WL–TR–96–2052

SOLUTION-ADAPTIVE CALCULATION OF UNSTEADY BLADE ROW INTERACTIONS IN TRANSONIC TURBOMACHINERY

Scott M. Richardson

Fan & Compressor Branch
Aero Propulsion and Power Directoriate

March 1996

Final Report for Period Apr 93 – Mar 96

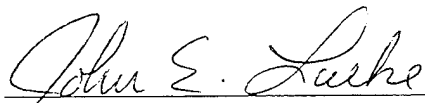APPROVED FOR PUBLIC RELEASE: DISTRIBUTION IS UNLIMITED

19960620 160

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government–related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foriegn nations.
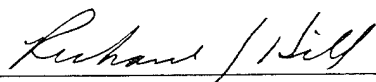
This technical report has been reviewed and is approved for publication.

JOHN E. LUEKE
Aerospace Engineer
Fan & Compressor Branch
Turbine Engine Division
Aero Propulsion and Power Directorate

MARVIN A. STIBICH
Chief
Fan & Compressor Branch
Turbine Engine Division
Aero Propulsion and Power Directorate

RICHARD J. HILL
Chief of Technology
Turbine Engine Division
Aero Propulsion and Power Directorate

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WL/POTF, WPAFB, OH 45433–7251 to help up maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractural obligations, or notice on a specific document.

	REPORT DOCUMENTATION PAGE		*Form Approved* *OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE MAR 1996	3. REPORT TYPE AND DATES COVERED FINAL APR 93 – MAR 96	
4. TITLE AND SUBTITLE SOLUTION–ADAPTIVE CALCULATION OF UNSTEADY BLADE ROW INTERACTIONS IN TRANSONIC TURBOMACHINERY			**5. FUNDING NUMBERS** PE 61102F PR 2307 TA S1 WU 27
6. AUTHOR(S) S. M. RICHARDSON			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AERO PROPULSION AND POWER DIRECTORATE WRIGHT LABORATORY AIR FORCE MATERIEL COMMAND WRIGHT-PATTERSON AFB 45433-7251			**8. PERFORMING ORGANIZATION REPORT NUMBER** WL-TR-96-2052
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AERO PROPULSION AND POWER DIRECTORATE WRIGHT LABORATORY AIR FORCE MATERIEL COMMAND WRIGHT-PATTERSON AFB OH 45433-7251			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER** WL-TR-96-2052

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE: DISTRIBUTION IS UNLIMITED	**12b. DISTRIBUTION CODE**

13. ABSTRACT *(Maximum 200 words)*

This report describes the development of an implicit, viscous method for the solution of the quasi-three-dimensional flow equations for rotor-stator interaction in transonic turbomachinery. The flow algorithm is described, followed by the implicit time-marching scheme, and the one-equation turbulence model. The algorithm is implemented on an unstructured grid arrangement of locally structured micro-blocks called "patches." Solution-dependent adaptation is used to refine the grid in regions containing flow features which require enhanced resolution. An overlapped sliding grid interface is used to transfer flow equation information between the respective blade grids.

The resulting computational algorithm has been used to perform a number of validation exercises and has been demonstrated on a modern transonic turbine stage. Where possible, these results are compared with experimental data and show the ability of the method to accurately capture the unsteady flow physics in a robust and computationally efficient manner.

14. SUBJECT TERMS COMPUTATIONAL FLUID DYNAMICS, ROTOR–STATOR INTERACTION, UNSTEADY FLOW			**15. NUMBER OF PAGES** 229
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	**18. SECURITY CLASSIFICATION OF THIS PAGE** UNCLASSIFIED	**19. SECURITY CLASSIFICATION OF ABSTRACT** UNCLASSIFIED	**20. LIMITATION OF ABSTRACT** SAR

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

ABSTRACT

Modern turbomachinery components are increasingly being designed to take into consideration the naturally unsteady flow environment existing within the machine. With the continued drive towards higher wheel speeds, increased blade loading, and closer blade row spacing, these unsteady effects have greater impact on engine performance. In order to understand these effects and account for them in the design process, accurate prediction and analysis of unsteady blade row interactions is required.

A method is presented for computing the unsteady blade row interaction flow field by solving the two-dimensional Navier-Stokes equations on a solution-adaptive unstructured mesh. The flow equations are discretized in a cell-vertex finite volume formulation and are solved with a Runge-Kutta algorithm used as the driver for a fully implicit scheme. The quasi-three-dimensional effects of streamtube contraction and radius change are included to allow calculation of modern turbomachinery designs. The effects of turbulence are included through the use of the one-equation Spalart-Allmaras model, allowing a prediction of the propagation of the turbulent eddy viscosity between blade rows. An overlapped, sliding interface is used between the separate blade row grids and is implemented in a conservative fashion to provide accurate communication of flow variable information between grids. The algorithm uses a series of locally structured micro-blocks, called "patches," which are arranged in a globally unstructured manner. Solution-dependent adaptation is used to refine these patches in regions containing flow features which require enhanced resolution.

The resulting computational algorithm has been used to perform a number of validation exercises and has been demonstrated on a modern transonic turbine stage. Where possible, these results are compared with experimental data and show the ability of the method to accurately capture the flow physics in a robust and computationally efficient manner.

Contents

List of Tables

List of Figures

NOMENCLATURE

English Symbols

A	cell area
A^+	sublayer thickness, 26
c	speed of sound or blade chord
c_{b1}, c_{b2}, c_{v1}, c_{w1}, c_{w2}, c_{w3}	turbulence model constants
c_p	constant pressure specific heat
c_v	constant volume specific heat
c_1, c_2, c_3, c_4	characteristic variables
C_{cp}	turbulence model constant, 1.6
C_D	drag coefficient
C_f	skin friction coefficient
C_{KLEB}	turbulence model constant, 0.3
C_L	lift coefficient
CFL	Courant-Friedrichs-Lewy stability limit
CFL^*	standard CFL limit without residual smoothing
D	cylinder diameter
$D(U)$	artificial dissipation operator
$D_t(U)$	time difference operator
e	specific internal energy

E	total internal energy
F, G	vector of inviscid fluxes
$\vec{F}$	body force vector
$\mathcal{F}$	convective flux
$g(z)$	amplification factor
h	streamtube thickness
H	source term vector or stagnation enthalpy
$\hat{i}_m,\ \hat{i}_\theta,\ \hat{i}_r$	unit vectors in streamwise, circumferential and radial directions
$j,\ k$	local cell coordinate directions
L	characteristic length for nondimensionalization
L_i	Lagrangian polynomial
$m,\ n,\ \theta$	coordinate system in streamwise, normal, and tangential directions
M	Mach number
$\hat{n}$	unit normal vector
Nu	Nusselt number
p	pressure or wave number
ΔP	pressure switch
Pr	Prandtl number, 0.72 for air
Pr_T	turbulent Prandtl number, 0.90
$\dot{q}_m,\ \dot{q}_\theta$	heat flux vector components
$\dot{Q}$	heat flux
r	streamtube radius
R	specific gas constant
R, S	vector of viscous fluxes
R_1, R_2, R_3, R_4	residuals
R_c	radius of curvature

$\mathcal{R}(U)$	residual
Re	Reynolds number, $\rho_\infty u_\infty c / \mu_\infty$
S	entropy function
S_V	arbitrary control volume surface
t	time
T	temperature
u	arbitrary velocity component
u^+	friction velocity, $(\tau_w / \rho_w)^{\frac{1}{2}}$
U	vector of dependent variables
$\hat{U}$	Fourier mode amplitude
v_m, v_θ	absolute velocity components in m, θ directions
V	arbitrary control volume
$\vec{V}, \vec{V}_r, \vec{V}_s,$	absolute, relative, and control surface velocity vectors
w_θ	relative velocity in θ direction, $v_\theta - r\Omega$
$\dot{W}$	work
x, y	Cartesian coordinates
y	distance from wall
y^+	dimensionless distance from the wall, yu^+/ν
z	axial direction
$z(\xi)$	Fourier symbol

Greek Symbols

α	flow angle
α_q, β_q	Runge-Kutta algorithm coefficients
β, β_j, β_k	artificial dissipation scaling coefficients

γ	specific heat ratio, 1.4 for air
δ^2, δ^4	second and fourth differences
$\Delta m_j,\ \Delta m_k,\ \Delta \theta_j,\ \Delta \theta_k$	average cell dimensions
Δt	real time step
Δt^*	cell time step with unit CFL number
$\Delta \tau$	pseudo-time step
$\epsilon, \epsilon_j, \epsilon_k$	implicit residual smoothing coefficients
ϵ_2, ϵ_4	second- and fourth-order numerical damping coefficients
η	surface normal direction
Θ	dilatation tensor
κ	thermal conductivity
κ_1	von Karman constant, 0.4
κ_2	Clauser's constant, 0.0168
λ	bulk viscosity, $-2\mu/3$, or 1-D CFL number
$\lambda_j,\ \lambda_k$	maximum eigenvalues in cell directions
$\mu,\ \mu_T$	molecular and turbulent viscosities
$\nu,\ \nu_T$	molecular and turbulent dynamic viscosities
$\tilde{\nu}$	turbulence model viscosity parameter
ξ	phase angle
$\overline{\overline{\Pi}}$	total stress tensor, $-p\delta_{ij} + \overline{\overline{\tau}}$
ρ	density
σ	turbulence model constant
$\sigma_2,\ \sigma_4$	second- and fourth-order numerical damping coefficients
τ	pseudo-time
$\overline{\overline{\tau}}$	viscous stress tensor
$\tau_{mm},\ \tau_{m\theta},\ \tau_{\theta\theta},\ \tau_{rr}$	viscous stress tensor components

Ω	blade row angular velocity
$\hat{\Omega}$	vorticity

Subscripts

c	cell center value
D	diameter
i	inviscid
L	plate length
max	maximum
n	node number
o	stagnation value
T	turbulent
v	viscous
w	wall value
x	axial direction
∞	freestream value

Superscripts

n	time step level
q	Runge-Kutta stage level

ACKNOWLEDGMENTS

Chapter 1

INTRODUCTION

Flow unsteadiness is the dominant aspect of flow through turbomachinery compo-
nents. Examples of this unsteadiness include turbulence effects, blade row interac-
tions, and rotating stall and surge. Of these, blade row interaction effects have the
greatest impact on the performance of both compressors and turbines. Standard
design practice ignores much of this unsteadiness, using steady-state analyses to
predict the time-averaged mean behavior of the flow. As design goals become more
aggressive, it becomes increasingly difficult to improve performance using methods
based on this time-averaged flow assumption. Therefore, a more sophisticated treat-
ment of the unsteady nature of the flow becomes vital. Both computational and
experimental resources need to be used to characterize, understand, and model these
effects to improve the capability and accuracy of current design tools. The ultimate
use for a computer code such as the one presented in this thesis is to provide an
insight into the physics of the interaction between moving blade rows.

1.1   Background

Current turbomachinery design methods use the S1-S2 streamsurface method first
described by Wu [80, 81], with two intersecting planes used to describe the three-
dimensional space within a turbomachinery blade passage (see Fig. 1.1). Typically,
an entire multi-stage turbomachinery compressor or turbine is designed along the
meridional $r - z$ plane (S1 surface) with blade-to-blade streamsurface (S2 surface)
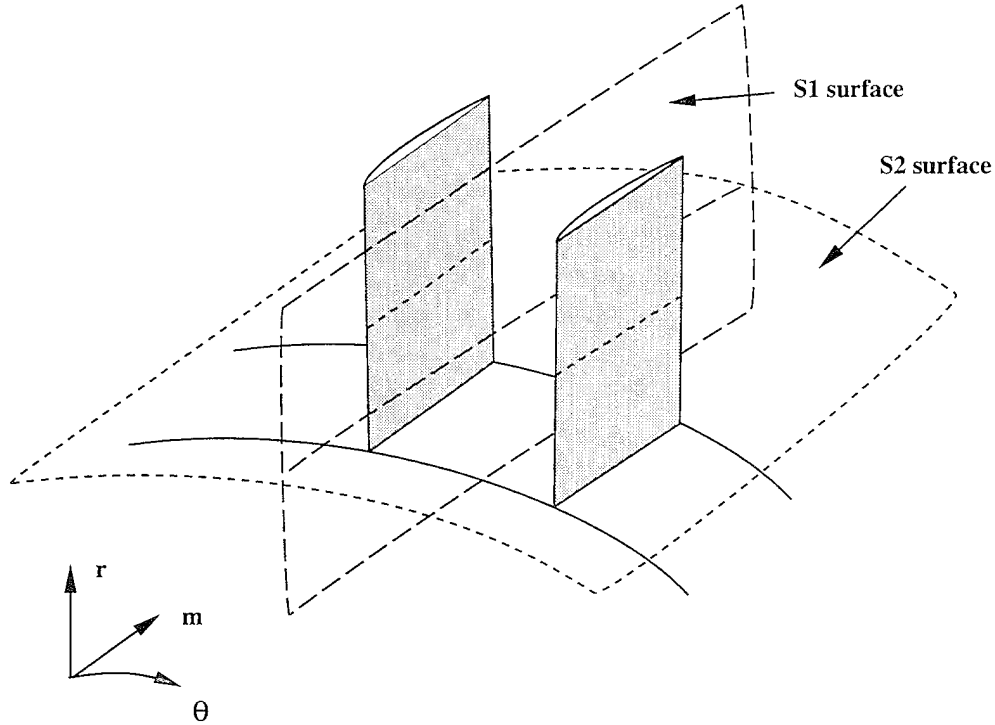
Figure 1.1: S1-S2 streamsurface model of Wu.

analyses used to improve the performance of the individual cascade sections. In order to perform the meridional analysis, each blade row is designed as an isolated cascade. The effect of the up- and downstream rows is only included as a circumferentially-averaged "force field." The obvious disadvantage of this approach is that it removes the impact of any of the natural flow unsteadiness from the design methodology.

An example of the impact of this unsteadiness may be found in transonic compressors. In this case, the position of the inlet shock pattern determines the mass flow through the blade row over a large portion of the operating line. The transonic compressor characteristic is typically very steep; small variations in mass flow may result in large changes in the operating pressure ratio. Therefore, unsteadiness in the shock position may cause the blade row to overflow resulting in a significant drop in the pressure ratio from the design point, as illustrated in Fig. 1.2. This effect has been observed experimentally by Fleeter et al. [25] during the test of a five-stage

compressor which had fully supersonic second and third stages. They observed an approximate 5% overflow in the second stage from the design value, with a large drop in machine pressure ratio.

Similar effects exist in transonic turbine designs. The two main concerns are the impingement of the vane trailing edge shock on the downstream rotor and the effect of unsteadiness on blade heat transfer. Both shock impingement, and the accumulation of low momentum wake fluid along the rotor pressure surface, can affect the heat transfer and reduce the effectiveness of the cooling flows added to protect the blade surfaces from the high temperatures of the turbine environment.

The purpose of this thesis is to present a numerical technique for modeling the unsteady interaction within modern turbomachinery components in a computationally cost effective manner. This work will discuss the development of the computer program, its implementation, and will present a demonstration of its performance on a modern turbine design.

1.2   Blade Row Interactions

Computationally, the blade row interaction problem has been studied using two major approaches. The first is to analyze only a single blade row while imposing inlet boundary conditions containing a representation of the blade wakes from an upstream blade row. Mitchell [49] and Hodson [31] used this approach to extend the inviscid time-marching method of Denton [24] to investigate the blade surface static pressure, and velocity amplitude and phase relationships, at the wake passing frequency. Giles [26] solved the unsteady Euler equations using a lagged periodicity boundary condition technique, which inclines the computational plane in time to allow calculation of arbitrary rotor/stator blade count ratios. Scott and Hankey [66]

3

Steady Wave Pattern

Unsteady Wave Pattern

Uniform Inlet
Veolocity Profile

Velocity Profile
Containing Wakes

Pressure
Ratio

Design Operating Point

Operating Point if
Downstream Stage
Overflows
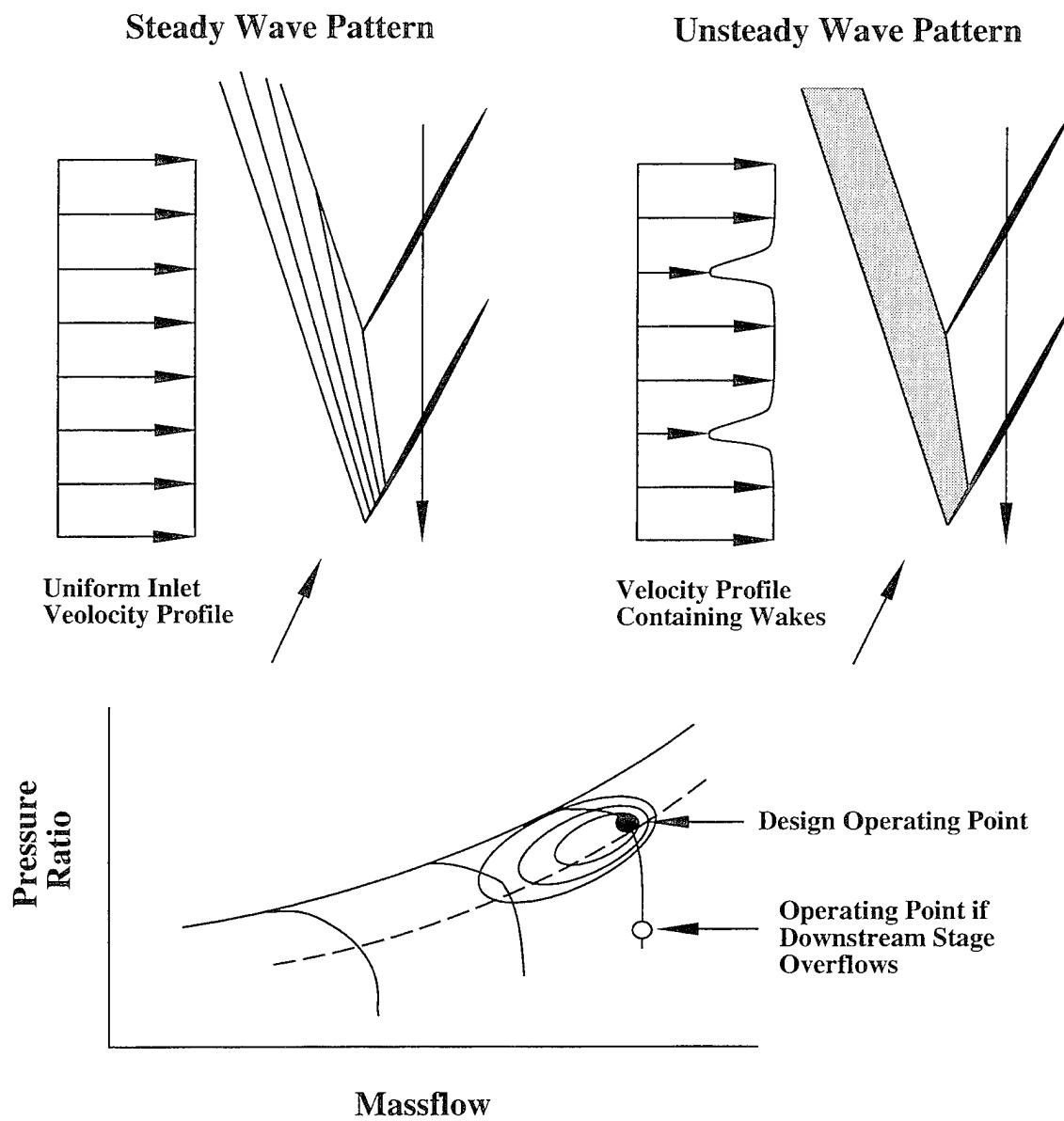
Massflow

Figure 1.2: Blade row interaction effect on transonic compressor design point.

4

and the present author [59] solved the Navier-Stokes equations in efforts to calculate the unsteady mass flow swallowing capacity of a compressor rotor cascade.

The second approach is to model both the rotor and stator blade rows as a fully coupled system, with the blade wakes generated by an upstream row impinging on the downstream blades. This approach has the added advantage that both shock impingement and the potential interaction between blades is modeled, as well as any changes in the upstream wake due to different stage loading conditions. The first two-dimensional calculations were done by Rai [52] using a third-order accurate implicit scheme with patched and overlaid grids to solve the thin-layer Navier-Stokes equations in a low-speed turbine stage. This method was later extended to compute a low-speed, multi-stage compressor by Gundy-Burlet et al. [30] and a multi-stage turbine by Lin and Yang [42]. Other two-dimensional codes have included the explicit codes of Jorgenson and Chima [36], Lewis et al. [41], Arnone, et al. [4, 5], and the present author [60]. Three-dimensional calculations were first presented by Rai [53, 54] for a low-speed turbine which included the effect of tip leakage over the rotor. Other prominent three-dimensional codes include the set of inviscid and viscous codes of Rao and Delaney [56] and Rao et al. [57, 58]. These methods all use multiple structured meshes, generally with patched or Chimera-type grid structures. Dawes [23] and Mathur et al. [44, 45] have recently developed unstructured codes to compute rotor-stator interactions, but neither of these methods includes adaptation of the grid to the unsteady flow field.

1.3 Numerical Technique

Generally, methods used for the blade-row interaction problem grid each blade row separately and then use a sliding interface to transfer information between the different blade rows. In the case of modern high-speed compressor and turbine designs,

the flow fields contain unsteady shock and wake patterns which cross this interface between grids. A large change in grid density across the interface will induce a significant dissipation of both wakes and shocks causing damping of the interaction effects. To adequately resolve the complex shock and wake structures using structured grids requires a considerably larger number of points than is typically used, most of them wasted in other, more benign, regions of the flow.

The use of unstructured grids is an alternative approach that removes many of the restrictions found in the structured grid approach. Virtually any grid point distribution can be used, allowing grid points to be added only where there is a need for increased resolution. Coupled with a solution-adaptive algorithm, unstructured grids allow the computational grid to evolve with the solution, reducing both computer time and storage requirements over those for a structured grid of adequate resolution. Recent literature has shown a number of examples where unstructured grids have been used for steady [61, 69] as well as unsteady [29, 44, 45] turbomachinery flows. There are few examples in the literature, however, where solution-adaptivity has been used in the computation of unsteady rotor-stator flows.

Another factor in the choice of a computational method is the selection of an explicit or implicit algorithm. Implicit solvers offer the benefit of larger time steps, which are particularly attractive for the highly stretched grids needed to adequately resolve viscous layers. For unsteady calculations, where small time steps are often required for adequate temporal resolution, the computational cost of an implicit solver used throughout the entire domain may overshadow the solver's ability to operate with larger time steps. Further, implicit solvers for unstructured meshes are computationally quite expensive. Jameson [32] and Weiss and Smith [76] have introduced similar techniques which use an explicit time marching method as the driver for a fully implicit scheme. These methods, which are adopted in this thesis, provide a considerable improvement over a purely explicit implementation. The

6

technique of Jameson has been applied to unsteady airfoil flows [32] and recently by Arnone for blade row interactions in turbomachinery [4, 5].

1.4 Turbulence Models

One of the requirements for the turbulence model used in this effort was that it should contain some form of the field equations to allow the propagation of the eddy viscosity from an upstream to a downstream blade row. A number of models are currently available, both in one- and two-equation form. Two-equation model choices include those of Jones and Launder [35], Coakley [16], Wilcox [79], and Menter [48]. Of these, the blended $k - \epsilon/k - \omega$ of Menter has perhaps the strongest record for the prediction of adverse pressure gradient flows. However, the additional computational overhead of a sixth field equation led to a consideration of the one-equation models of Johnson and King [34], Baldwin and Barth [8], and Spalart and Allmaras [71]. The Spalart and Allmaras model was chosen for the current effort; it has the advantage of being a local model, easily implemented in an unstructured grid environment, and is particularly well suited to predictions with coarser grids.

1.5 Adaptation Methods

Solution-adaptation techniques are generally classified into three basic categories. The first, and one that may be applied to either structured or unstructured meshes, is mesh movement in response to flow field topology [11, 20]. Original mesh connectivity is maintained, with points moving towards or away from regions of high or low gradients, respectively. The obvious disadvantage of this approach is the restriction to the original number of mesh points and their original connectivity. Complex flow features may rob the remaining mesh of resolution, resulting in an overall grid distribution which is unsatisfactory.

The second approach, and the one primarily used with triangular element meshes, is mesh regeneration, where the computational mesh is completely renewed during an adaptation cycle [50]. One advantage of this approach is that no change is required to the flow solver since the logical mesh structure is not modified after refinement. Additionally, re-meshing gives superior control over variations in grid cell size and allows alignment of the grid with flow features. The primary drawback of this method is that the computational time required to perform the re-meshing may be prohibitive, particularly when the mesh must be redefined repeatedly as in the case of a rotor-stator interaction type problem.

The third choice, and the one used in the current solver, is grid enrichment through local cell division and recombination, the technique most commonly used with quadrilateral meshes. Here the primary advantage is the speed at which the grid may be adapted, of prime importance in the computation of unsteady flows. Special coding logic is required to account for the midface nodes created at the interface between divided and undivided cells. A series of rules is applied during the adaptation to insure that cells divide in the proper sequence to inhibit the formation of cells with multiple midface nodes on a side. Triangular cell unstructured mesh methods may also use local cell refinement, but the refinement must be followed by a local retriangulation to insure a smooth mesh [9]. It is unclear from the literature whether there is a performance advantage between quadrilateral and triangular grid enrichment techniques.

1.6 Present Research

The objective of the current study is the development of an unsteady solution-adaptive unstructured grid method for application to blade row interactions in transonic turbomachinery. While the method is primarily for axial cascades, the equa-

8

tions are cast in the more general quasi-three-dimensional formulation to include the effects of radius and streamtube thickness variation found in centrifugal and mixed flow designs.

The algorithm used to perform the time integration of the flow equations uses a modification of the implicit reformulation of Jameson [32], allowing time accurate computations and the use of convergence acceleration techniques. The effects of turbulence is modeled with the either the algebraic model of Baldwin and Lomax [7] or the one-equation model of Spalart and Allmaras [71].

Results are presented for a variety of test cases used to separately validate the various components of the computer code. The fully unsteady method is demonstrated, both with and without grid adaptation, with the calculation of a transonic turbine design.

1.7   Retrospective

The work described in this thesis is the culmination of approximately ten years of effort in the area of modeling of unsteady flows in turbomachinery. This section describes a number of the directions pursued and attempts to summarize the various methods and the motivation for their use.

The author's earliest work in this area was the use of modified version of the highly successful structured code developed by Shang et al. [67, 68], which is based on the explicit MacCormack predictor-corrector method [43]. This code was used by this author [59], as well as by Scott and Hankey [66], to predict the unsteady mass flow swallowing capacity of a transonic fan cascade using an unsteady inlet wake boundary condition. Although these efforts did show an effect of the upstream wake parameters on the time averaged mass flow, questions remained on the validity of the wake boundary condition. Additionally, the potential flow interaction between

blade rows was not included in the analysis. These considerations led to a further modification of the structured code to predict a fully interactive rotor-stator flow field.

While this code was successful in computing, to the author's knowledge, the first prediction of rotor-stator interaction in a transonic compressor [60], it suffered from a number of deficiencies. The first of these was the use of a grid with inadequate density. When computing an isolated airfoil or cascade flow, the usual practice is to stretch the grid near the blade surface and downstream of the trailing edge in order to adequately resolve the large velocity gradients in the blade boundary layer and wake regions. Typically, the remainder of the flow domain is relatively sparsely gridded because the gradients, except in the case of shocks, are considerably more mild. Correct resolution of the blade wake is of particular importance in turbomachinery cascades for proper prediction of cascade losses in both compressors and turbines, as well as the blade heat transfer in turbines. Shock resolution has also received considerable attention in external airflow calculations, with airfoil grids often clustered in the regions of expected shocks, in an effort to improve the resolution in the shock-boundary layer interaction region. In turbomachinery, shocks play a considerably larger role; setting the mass flow swallowing capacity of transonic compressors, and impacting transonic turbine heat transfer and aeromechanical response.

Including the propagation of upstream blade wakes into a downstream row is important for the modeling of the unsteady flow field. To accomplish this requires that the grid resolution is maintained between up- and downstream rows. Since the downstream row is moving relative to the upstream, the circumferential grid resolution in the downstream row would need to everywhere match the *finest* resolution in the upstream row. A similar requirement exists for the grid clustering needed around the shocks that propagate between blade rows. The need for significantly enhanced grid resolution obviously poses a considerable challenge to both memory and com-

puting time and was the prime driver behind the selection of a solution-adaptive unstructured grid method.

Another issue in the original rotor-stator code was the use of an algebraic turbulence model, the industry standard at the time. An algebraic model computes the turbulent eddy viscosity by searching away from the wall surface to determine which type of profile model to apply. This search is often computationally expensive in unstructured grids, which do not have the natural connectivity between grid points contained in structured grids. More important however, any method to model the propagation of the the turbulent eddy viscosity from one blade row to another would be woefully inadequate using an algebraic model. Therefore, it was felt that any turbulence model used for rotor-stator calculations must include some form of the field equations in order to capture the effect of turbulent eddy viscosity convection and diffusion. An added difficulty with the algebraic model was its extremely poor performance in the regions of strong pressure gradient often found in modern, highly loaded, transonic compressors. Calculations showed an unrealistically large region of separation, causing a large wake which would create a "subsonic gap" in the flow entering the rotor. The unsteady animation of this flow showed a section of the shock wave propagating forward inside the low-speed flow of the stator wake, triggering an axial oscillation of the stator separation point. These computations clearly showed a secondary pulsation of the flow caused by the shock induced unsteadiness within the wake. While quite entertaining to watch, there is little doubt that the inadequacies of the turbulence model were largely responsible for this phenomenon. An ability to properly account for strong pressure gradients is therefore also included in the turbulence model selection criteria.

1.8 Overview of Thesis

In Chapter 2, the Navier-Stokes equations are derived in a general sense, including a description of all assumptions required for the present effort. Using the more detailed analysis included in Appendix A, these equations are presented in the specific quasi-3D form, which includes the effects of radius and streamtube thickness variation. The method of incorporating these distributions is then examined, followed by an explanation of the nondimensionalization of the flow equations.

Chapter 3 describes the finite volume solution procedure implemented on an unstructured mesh of quadrilateral cells. The method used for the evaluation of the inviscid and viscous terms is presented, with a discussion of the boundary conditions and the inclusion of artificial dissipation. The dual time stepping algorithm and convergence acceleration techniques are described, followed by an examination of the numerical properties of the scheme. The chapter is completed with a description of the implementation of the computer program and its semi-structured grid approach.

Chapter 4 presents a discussion of the two turbulence models available in the code. The simple algebraic model of Baldwin and Lomax [7] is first described, followed by the one-equation model of Spalart and Allmaras [71]. A discussion of the special requirements of the unstructured grid implementations of both methods is presented.

In Chapter 5 the method used for the generation and adaptation of the unstructured grid is described. Cell division and recombination is used to either add or remove individual cells from the mesh using the local flow gradients to drive the addition or removal of mesh points. Adaptation of the grid to both strong and weak flow features is described, followed by an examination of the numerical treatment of the midface nodes.

Chapter 6 presents the results of the numerical experiments performed for this effort. Validation cases are presented which exercise various portions of the code independently, followed by a sample transonic turbine application. A number of turbine cases are presented to highlight the consequences of the various parameter choices available in the computation of unsteady flows.

Finally, Chapter 7 presents the major conclusions of this work. This chapter also includes a discussion of the limitations of this effort and recommendations for future work.

The thesis is completed with three appendices. The first presents a full derivation of the quasi-3D form of the flow equations. The second contains a detailed description of the boundary conditions implemented in the computer program. The interactive grid generator used to construct the computational grids is described in the final appendix.

Chapter 2

EQUATIONS OF MOTION

In this chapter the general Navier-Stokes equations are presented, including the assumptions that are required for their derivation. The quasi-three-dimensional or quasi-3D formulation of these equations is then given and expressed in a Reynolds averaged, nondimensional form.

2.1 Navier-Stokes Equations

Consider the two-dimensional representation of an arbitrary control volume V as shown in Fig. 2.1. This volume is bounded by surface S_V, with outward normal vector $\hat{n}$. If this bounding surface moves with velocity $\vec{V}_s$ and the fluid velocity relative to the surface is $\vec{V}_r$, then the absolute velocity of the fluid is given by $\vec{V} = \vec{V}_s + \vec{V}_r$.

Conservation of mass within this control volume requires that the time rate-of-change of mass within the volume is balanced by the net flux of mass across the bounding surfaces, i.e.:

$$\frac{d}{dt} \int \int \int_V \rho dV + \oint_{S_V} \rho \vec{V}_r \cdot \hat{n} dS = 0. \tag{2.1}$$

This equation is based on the assumption that the fluid is continuous and that all flow variables are continuous functions in space.

Newton's second law requires that the time rate-of-change of momentum within the volume, plus the momentum flux convected through the boundary, is balanced
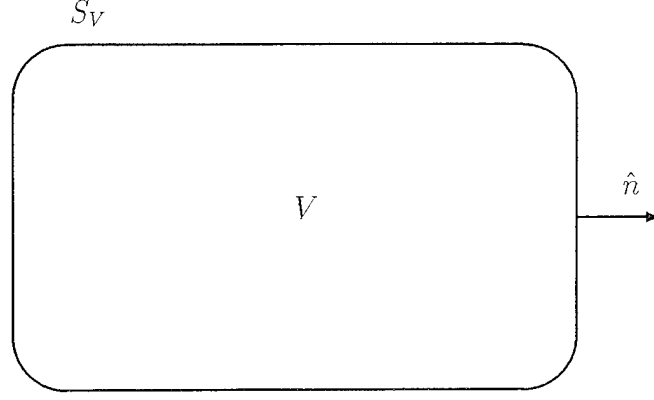
Figure 2.1: Control volume for conservation equations.

by the sum of all forces acting on the boundary of the control volume. This is expressed as:

$$\frac{d}{dt}\int\int\int_V \rho\vec{V_r}dV + \oint_{S_V} \rho\vec{V}(\vec{V_r}\cdot\hat{n})dS = \Sigma\vec{F}. \qquad (2.2)$$

Neglecting body force terms, the forces acting on the boundary are expressed as a summation of pressure and shear stress terms:

$$\Sigma\vec{F} = \oint_{S_V} (-p + \overline{\overline{\tau}})\cdot\hat{n}dS, \qquad (2.3)$$

where p is the local static pressure and $\overline{\overline{\tau}}$ is the shear stress tensor.

The first law of thermodynamics states that the time rate-of-change of the total energy E within the control volume, plus the energy flux convected through the boundary, is balanced by the sum of the heat flux $\dot{Q}$ and work $\dot{W}$ done by the stresses on the fluid. This relation is given by:

$$\frac{d}{dt}\int\int\int_V EdV + \oint_{S_V} E(\vec{V_r}\cdot\hat{n})dS = \dot{Q} - \dot{W}. \qquad (2.4)$$

16

Where $\dot{Q}$ includes heat conduction alone, with the assumption that there are no heat sources internal to the control volume. The work $\dot{W}$ is given by:

$$\dot{W} = \oint_{S_V} \left(\overline{\overline{\tau}} \vec{V} - p\vec{V} \right) \cdot \hat{n} dS. \tag{2.5}$$

Auxiliary relationships are required to close the above system of equations. The first is the assumption that the working fluid is a thermally perfect gas:

$$p = \rho RT, \tag{2.6}$$

where R is the gas constant. Further, the gas is assumed to be calorically perfect (i.e., with constant specific heats c_v and c_p). The specific internal energy of the gas is then defined as:

$$e = c_v T. \tag{2.7}$$

Using the relationship between the static pressure p and the total energy of the fluid E, the above equations can be combined to yield:

$$E = \rho e + \frac{\rho}{2} \left(v_m^2 + v_\theta^2 \right) \tag{2.8}$$

and

$$p = (\gamma - 1) \left[E - \frac{\rho}{2} \left(v_m^2 + v_\theta^2 \right) \right], \tag{2.9}$$

where γ is the ratio of specific heats, and

$$\vec{V} = v_m \hat{i}_m + v_\theta \hat{i}_\theta. \tag{2.10}$$

17

The viscosity of the fluid μ is expressed as a function of temperature using Sutherland's Law:

$$\frac{\mu}{\mu_\infty} = \left(\frac{T}{T_\infty}\right)^{\frac{3}{2}} \frac{(T_\infty + T_{ref})}{(T + T_{ref})}, \tag{2.11}$$

where the reference temperature $T_{ref} = 110.4K$ for air, and μ and μ_∞ are the viscosities at temperatures T and T_∞.

Finally, the thermal conductivity of the working fluid κ is specified using the Prandtl number:

$$Pr = \frac{\mu c_p}{\kappa}. \tag{2.12}$$

A value of $Pr = 0.72$ is normally taken for air at standard conditions.

2.2 Quasi-3D Formulation

Until this point in the derivation, an arbitrary three-dimensional control volume has been used. The specific control volume used to produce the equations in their final quasi-3D form must now be described. The configuration of interest is the computation of the blade-to-blade flow in axial and radial turbomachinery with the inclusion of radius and streamsheet thickness variations. Therefore, a two-dimensional axisymmetric coordinate system (m, θ) is used, where m is the streamwise direction and θ is the cylindrical coordinate rotating with the blade row:

$$dm^2 = dz^2 + dr^2, \tag{2.13}$$

where z is the axial direction, and

$$\theta = \theta' - \Omega t. \tag{2.14}$$
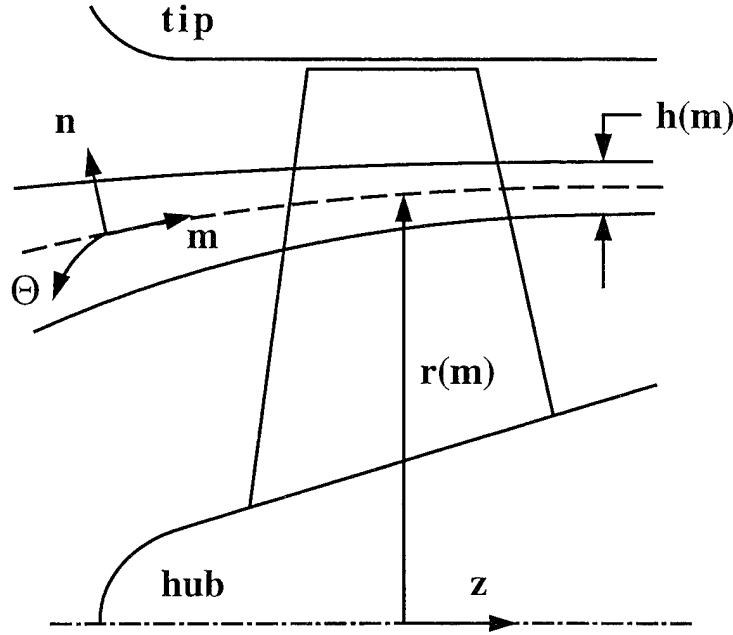
Figure 2.2: Quasi-3D streamsurface through a typical fan rotor.

Here θ' is fixed in space and Ω is the angular velocity of the blade row. Additionally, the velocity normal to the streamtube lamina is assumed to be zero. The streamsurface for this coordinate system through a typical fan rotor is shown in Fig. 2.2. The computational domain is a two-dimensional blade-to-blade surface discretized with a grid of computational elements or cells. These cells have a thickness and radial position which is accounted for in the quasi-3D formulation.

In this coordinate system, the radius r and the streamtube thickness h are taken to be known functions of m. The equations presented below reduce to the standard two-dimensional Navier-Stokes equations in conservative form when r and h are constants. The validation cases presented later are computed with $r = h = 1$, while the cascade computations are performed with radius and streamtube thickness variations defined from experimental or design data.

The Navier-Stokes equations for this domain may be expressed in an integral formulation, which is used in a finite volume implementation as follows:

$$\frac{d}{dt} \iint_{\substack{cell \\ area}} U\,dm\,d\theta + \oint_{\substack{cell \\ faces}} \left(F\,d\theta - G\,dm \right) =$$
$$\iint_{\substack{cell \\ area}} H\,dm\,d\theta + \oint_{\substack{cell \\ faces}} \left(R\,d\theta - S\,dm \right). \tag{2.15}$$

Here, t is time, U is the dependent variable vector, F and G are the inviscid flux vectors, R and S are the viscous flux vectors, and H is the source term vector incorporating the additional radius and streamtube thickness variation terms. The full derivation of the quasi-3D equations is presented in Appendix A. The resulting vector components are given by (using the notation of Chima [15]):

$$U = rh \begin{bmatrix} \rho \\ \rho v_m \\ \rho v_\theta r \\ E \end{bmatrix}, \quad F = rh \begin{bmatrix} \rho v_m \\ \rho v_m^2 + p \\ (\rho v_m v_\theta) r \\ v_m(E + p) \end{bmatrix},$$

$$G = h \begin{bmatrix} \rho w_\theta \\ \rho w_\theta v_m \\ (\rho w_\theta v_\theta + p) r \\ w_\theta(E + p) + r\Omega p \end{bmatrix}, \quad H = rh \begin{bmatrix} 0 \\ H_2 \\ 0 \\ 0 \end{bmatrix},$$

$$R = rh \begin{bmatrix} 0 \\ \tau_{mm} \\ \tau_{m\theta} r \\ v_m \tau_{mm} + v_\theta \tau_{m\theta} - \dot{q}_m \end{bmatrix}, \quad S = h \begin{bmatrix} 0 \\ \tau_{m\theta} \\ \tau_{\theta\theta} r \\ v_m \tau_{m\theta} + v_\theta \tau_{\theta\theta} - \dot{q}_\theta \end{bmatrix}. \tag{2.16}$$

The source term is given by:

$$H_2 = \left(\rho v_\theta^2 + p - \tau_{\theta\theta}\right)\frac{r_m}{r} + (p - \tau_{rr})\frac{h_m}{h},$$

using the notation

$$\frac{r_m}{r} = \frac{1}{r}\frac{dr}{dm}, \qquad \frac{h_m}{h} = \frac{1}{h}\frac{dh}{dm}.$$

The static pressure is given by:

$$p = (\gamma - 1)\left[E - \frac{\rho}{2}\left(v_m^2 + v_\theta^2\right)\right], \tag{2.17}$$

and the stress and heat conduction terms are:

$$\tau_{mm} = 2\mu\frac{\partial v_m}{\partial m} + \lambda\Theta,$$

$$\tau_{\theta\theta} = 2\mu\left(\frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m\frac{r_m}{r}\right) + \lambda\Theta,$$

$$\tau_{m\theta} = \mu\left(\frac{\partial v_\theta}{\partial m} + \frac{1}{r}\frac{\partial v_m}{\partial \theta} - v_\theta\frac{r_m}{r}\right),$$

$$\tau_{rr} = 2\mu v_m\frac{h_m}{h} + \lambda\Theta,$$

$$\dot{q}_m = \kappa\frac{\partial T}{\partial m}, \qquad \dot{q}_\theta = \kappa\frac{1}{r}\frac{\partial T}{\partial \theta}, \tag{2.18}$$

where $w_\theta = v_\theta - r\Omega$ is the relative velocity, $\lambda = -2\mu/3$, and the dilatation is given by:

$$\Theta = \left[\frac{\partial v_m}{\partial m} + \frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m\left(\frac{r_m}{r} + \frac{h_m}{h}\right)\right].$$

2.3   Reynolds Averaging

Flows in modern turbomachinery components are highly turbulent due to the high Reynolds numbers of the flows. Turbulence is characterized by flow unsteadiness over

a wide range of time and length scales, the resolution of which is quite impractical for actual turbomachinery geometries using current computing resources. Instead, the high frequency unsteadiness of turbulent flow is modeled by assuming that all flow variables are comprised of two components. The first is a value which is Reynolds averaged over a given time interval and the second component which fluctuates about this average to represent the turbulence. The Reynolds averaging (or more correctly, mass, or Favre averaging for compressible flows) results in the addition of apparent stress and heat transfer terms to the momentum and energy equations. Generally, the Bussinesq approximation is used, which assumes that these additional terms are proportional to the mean strain rate and temperature gradient, respectively. In essence, this modeling takes the form of a redefinition of the viscosity and thermal conductivity:

$$\mu_{eff} = \mu + \mu_T, \quad \kappa_{eff} = \kappa + \kappa_T \tag{2.19}$$

where μ and κ represent the molecular values of the viscosity and thermal conductivity and μ_T and κ_T are their turbulent counterparts. The turbulent Prandtl number

$$Pr_T = \frac{\mu_T c_p}{\kappa_T} \tag{2.20}$$

is assumed constant, with a value of $Pr_T = 0.90$. The final form of the Reynolds averaged stress and heat transfer terms is given by:

$$\tau_{mm} = \frac{2}{3}(\mu + \mu_T) \left[2\frac{\partial v_m}{\partial m} - \frac{1}{r}\frac{\partial v_\theta}{\partial \theta} - v_m \left(\frac{r_m}{r} + \frac{h_m}{h} \right) \right],$$

$$\tau_{\theta\theta} = \frac{2}{3}(\mu + \mu_T) \left[2\frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m \left(2\frac{r_m}{r} - \frac{h_m}{h} \right) - \frac{\partial v_m}{\partial m} \right],$$

$$\tau_{m\theta} = (\mu + \mu_T) \left[\frac{\partial v_\theta}{\partial m} + \frac{1}{r}\frac{\partial v_m}{\partial \theta} - v_\theta \frac{r_m}{r} \right],$$

Variable	Reference	Normalized Freestream Value
ρ	ρ_∞	1
v_m, v_θ	u_∞	$\cos(\alpha_\infty), \sin(\alpha_\infty)$
p	$\rho_\infty u_\infty^2$	$\dfrac{1}{\gamma M_\infty^2}$
T	T_∞	1
E	$\rho_\infty u_\infty^2$	$\dfrac{1}{\gamma(\gamma-1)M_\infty^2} + 0.5$
μ	$\mu_\infty(T_\infty)$	1
r, m, h	L	
t	L/u_∞	
Ω	u_∞/L	

Table 2.1: Summary of nondimensional reference quantities.

$$\tau_{rr} = \frac{2}{3}(\mu + \mu_T)\left(v_m\left(2\frac{h_m}{h} - \frac{r_m}{r}\right) - \frac{\partial v_m}{\partial m} - \frac{1}{r}\frac{\partial v_\theta}{\partial \theta}\right), \qquad (2.21)$$

and

$$\dot{q}_m = -\left(\frac{\mu}{Pr} + \frac{\mu_T}{Pr_T}\right)\frac{\partial T}{\partial m},$$

$$\dot{q}_\theta = -\left(\frac{\mu}{Pr} + \frac{\mu_T}{Pr_T}\right)\frac{1}{r}\frac{\partial T}{\partial \theta}. \qquad (2.22)$$

The models used to evaluate the turbulent eddy viscosity μ_T are presented in further detail in Chapter 4.

2.4   Nondimensionalization

These equations are made dimensionless by selection of a characteristic length L, inlet freestream static density ρ_∞, inlet velocity u_∞, inlet static temperature T_∞, and molecular viscosity $\mu_\infty(T_\infty)$. Table 2.1 gives a summary of the reference quantities used for nondimensionalization. As a notational convenience, all variables will henceforth be assumed to represent their nondimensional equivalents.

23

The only effects of the nondimensionalization are in the viscous stress and heat conduction terms which now become:

$$\tau_{mm} = \frac{2}{3}(\mu + \mu_T)\frac{1}{Re}\left(2\frac{\partial v_m}{\partial m} - \frac{1}{r}\frac{\partial v_\theta}{\partial \theta} - v_m\left(\frac{r_m}{r} + \frac{h_m}{h}\right)\right),$$

$$\tau_{\theta\theta} = \frac{2}{3}(\mu + \mu_T)\frac{1}{Re}\left(2\frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m\left(2\frac{r_m}{r} - \frac{h_m}{h}\right) - \frac{\partial v_m}{\partial m}\right),$$

$$\tau_{m\theta} = (\mu + \mu_T)\frac{1}{Re}\left(\frac{\partial v_\theta}{\partial m} + \frac{1}{r}\frac{\partial v_m}{\partial \theta} - v_\theta\frac{r_m}{r}\right),$$

$$\tau_{rr} = \frac{2}{3}(\mu + \mu_T)\frac{1}{Re}\left(v_m\left(2\frac{h_m}{h} - \frac{r_m}{r}\right) - \frac{\partial v_m}{\partial m} - \frac{1}{r}\frac{\partial v_\theta}{\partial \theta}\right),$$

$$\dot{q}_m = -\frac{1}{(\gamma - 1)ReM_\infty^2}\left(\frac{\mu}{Pr} + \frac{\mu_T}{Pr_T}\right)\frac{\partial T}{\partial m},$$

$$\dot{q}_\theta = -\frac{1}{(\gamma - 1)ReM_\infty^2}\left(\frac{\mu}{Pr} + \frac{\mu_T}{Pr_T}\right)\frac{1}{r}\frac{\partial T}{\partial \theta}, \qquad (2.23)$$

where the Reynolds number is:

$$Re = \frac{\rho u_\infty L}{\mu_\infty}. \qquad (2.24)$$

The characteristic length L is typically taken as a unit length in the coordinate system. The Mach and Reynolds numbers, M_∞ and Re, and the inlet flow angle α_∞ are specified through the boundary conditions.

24

Chapter 3

UNSTRUCTURED GRID ALGORITHM

This chapter contains a description of the numerical method used in the current effort and details of its implementation. A description is provided for the discretization of the inviscid and viscous terms, and the boundary conditions employed. Implementation details of the artificial dissipation operator and the numerical integration scheme are given next, followed by an examination of the numerical properties of the overall unstructured scheme. The chapter concludes with a discussion of the semi-structured implementation strategy and the resulting data structures used in the computer code.

3.1 Inviscid Terms

A one-step Lax Wendroff-style integration scheme is used to discretize the convective terms in the inviscid portion of Eqn. 2.15 on a stationary grid of quadrilateral cells. Considering a single cell, as in Fig. 3.1, the equivalent integration scheme becomes:

$$\frac{d}{dt} \int\int_{\substack{cell \\ area}} U dm d\theta + \oint_{\substack{cell \\ faces}} (F d\theta - G dm) = \int\int_{\substack{cell \\ area}} H_i dm d\theta \qquad (3.1)$$

where U is the state vector, F and G are the convective flux vectors, and H_i is the inviscid portion of the source term vector H. Physically, the first term represents the time rate-of-change of the state vector U over a cell with area A, discretized as $\frac{\Delta U_c}{\Delta t} A$. The subscript c indicates the cell center and $\Delta U_c = U_c^{n+1} - U_c^n$ is the change

25
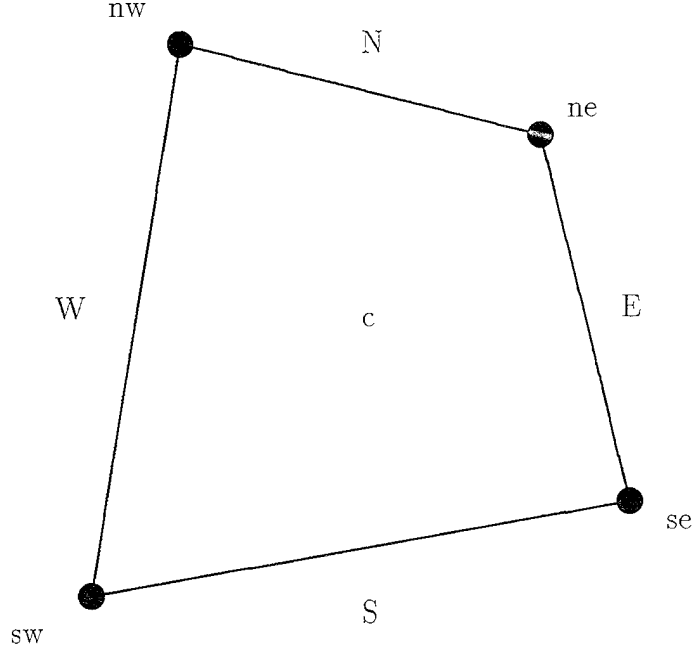
Figure 3.1: Quadrilateral cell node and face designations.

in U_c over a single time step. The second term represents the contribution of the convective fluxes and is calculated using trapezoidal integration. For example, the convective flux on the west face of the cell in Fig. 3.1 is given by:

$$\left(\frac{F_{nw} + F_{sw}}{2}\right)(\theta_{nw} - \theta_{sw}) - \left(\frac{G_{nw} + G_{sw}}{2}\right)(m_{nw} - m_{sw})$$

where the subscripts on F and G indicate the value of the fluxes at these corners. Finally, the third term represents the contribution of the inviscid source term over the cell area, discretized as $\Delta H_{ic}A$. The complete discretization of Eqn. 3.1 is:

$$
\begin{aligned}
\frac{\Delta U_c}{\Delta t}A \quad & + \quad \left(\frac{F_{nw} + F_{sw}}{2}\right)(\theta_{nw} - \theta_{sw}) - \left(\frac{G_{nw} + G_{sw}}{2}\right)(m_{nw} - m_{sw}) \\
& + \quad \left(\frac{F_{ne} + F_{nw}}{2}\right)(\theta_{ne} - \theta_{nw}) - \left(\frac{G_{ne} + G_{nw}}{2}\right)(m_{ne} - m_{nw}) \\
& + \quad \left(\frac{F_{se} + F_{ne}}{2}\right)(\theta_{se} - \theta_{ne}) - \left(\frac{G_{se} + G_{ne}}{2}\right)(m_{se} - m_{ne}) \qquad (3.2)\\
& + \quad \left(\frac{F_{sw} + F_{se}}{2}\right)(\theta_{sw} - \theta_{se}) - \left(\frac{G_{sw} + G_{se}}{2}\right)(m_{sw} - m_{se}) = \Delta H_{ic}A,
\end{aligned}
$$

26

where the cell area A is given as:

$$A = \frac{1}{2}\left[(m_{ne} - m_{sw})(\theta_{nw} - \theta_{se}) - (m_{nw} - m_{sc})(\theta_{ne} - \theta_{sw})\right]. \tag{3.3}$$

It should be noted that A as given above is not a true area, but with the inclusion of the r and h terms in the state and flux vectors, forms an equivalent cell volume in the quasi-3D coordinate system.

3.2   Viscous Terms

Considering the contributions of the inviscid and viscous portions of the equations separately, the complete change in the dependent variables may be written:

$$\frac{d}{dt}\iint_{\substack{cell\\area}} U\,dm\,d\theta = \frac{d}{dt}\iint_{\substack{cell\\area}} U_i\,dm\,d\theta + \frac{d}{dt}\iint_{\substack{cell\\area}} U_v\,dm\,d\theta, \tag{3.4}$$

where the two terms on the right hand side of Eqn. 3.4 represent the inviscid and viscous contributions to the time rate-of-change of the state vector U, respectively. Again using a stationary grid of quadrilateral cells, the integration of the viscous terms takes the form:

$$\frac{d}{dt}\iint_{\substack{cell\\area}} U_v\,dm\,d\theta = \oint_{\substack{cell\\faces}} (R\,d\theta - S\,dm) + \iint_{\substack{cell\\area}} H_v\,dm\,d\theta. \tag{3.5}$$

Here the second term represents the contribution of the diffusive fluxes, and H_v is the viscous portion of the source term.

The control volume used for the viscous terms is modified slightly from that used for the inviscid terms. The aim is to write the second derivatives in the viscous terms in such a manner that the complete second derivative at each node point can be assembled solely from the information available in the surrounding cells. As is
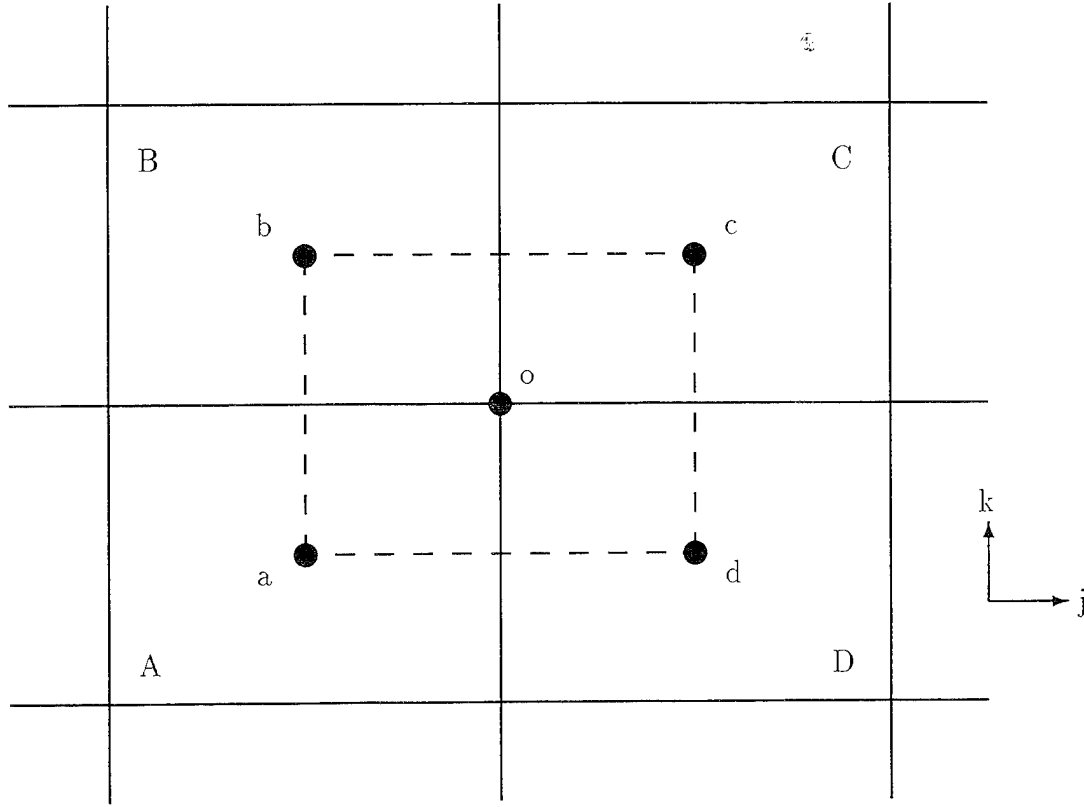
Figure 3.2: Primary cell for integration viscous terms.

the case for the inviscid terms described above, these derivatives are not complete until a sweep through all cells has been accomplished.

Considering a control volume around point o, as shown in Fig. 3.2, the line integration about the cell $abcd$ may be written (omitting the source term for clarity):

$$
\begin{aligned}
\Delta U_{v_o} &= \frac{\Delta t_{abcd}}{A_{abcd}} \left(\oint_{\substack{cell \\ faces}} (R d\theta - S dm) \right) \\
&= \frac{\Delta t_{abcd}}{A_{abcd}} \left\{ \left(\frac{R_d + R_a}{2} \right) (\theta_d - \theta_a) - \left(\frac{S_d + S_a}{2} \right) (m_d - m_a) \right. \\
&\quad + \left(\frac{R_c + R_d}{2} \right) (\theta_c - \theta_d) - \left(\frac{S_c + S_d}{2} \right) (m_c - m_d) \\
&\quad + \left(\frac{R_b + R_c}{2} \right) (\theta_b - \theta_c) - \left(\frac{S_b + S_c}{2} \right) (m_b - m_c) \\
&\quad \left. + \left(\frac{R_a + R_b}{2} \right) (\theta_a - \theta_b) - \left(\frac{S_a + S_b}{2} \right) (m_a - m_b) \right\}
\end{aligned}
$$

$$= \frac{\Delta t_{abcd}}{A_{abcd}} \left\{ \bar{R}_{da}\Delta\theta_{da} - \bar{S}_{da}\Delta m_{da} + \bar{R}_{cd}\Delta\theta_{cd} - \bar{S}_{cd}\Delta m_{cd} \right.$$

$$\left. + \bar{R}_{bc}\Delta\theta_{bc} - \bar{S}_{bc}\Delta m_{bc} + \bar{R}_{ab}\Delta\theta_{ab} - \bar{S}_{ab}\Delta m_{ab} \right\}, \qquad (3.6)$$

where $\bar{R}_{da} = \frac{1}{2}(R_d + R_a)$, $\bar{S}_{da} = \frac{1}{2}(S_d + S_a)$, etc. and $\Delta\theta_{da} = \theta_d - \theta_a$, $\Delta m_{da} = m_d - m_a$, etc.

In order to rewrite this equation in terms of cell variables, the following geometric terms are defined. Considering the generic cell shown in Fig. 3.1, with the local cell grid coordinates j and k as shown, define:

$$\Delta m_j = \frac{1}{2}(m_{ne} + m_{se} - m_{nw} - m_{sw}),$$

$$\Delta\theta_j = \frac{1}{2}(\theta_{ne} + \theta_{se} - \theta_{nw} - \theta_{sw}),$$

$$\Delta m_k = \frac{1}{2}(m_{ne} + m_{nw} - m_{se} - m_{sw}),$$

$$\Delta\theta_k = \frac{1}{2}(\theta_{ne} + \theta_{nw} - \theta_{se} - \theta_{sw}). \qquad (3.7)$$

Noting that $\Delta\theta_{da} = \frac{1}{2}(\Delta\theta_{j_D} + \Delta\theta_{j_C})$, $\Delta\theta_{cd} = \frac{1}{2}(\Delta\theta_{k_C} + \Delta\theta_{k_D})$, etc., Eqn. 3.6 becomes:

$$\Delta U_{v_o} = \frac{\Delta t_{abcd}}{2A_{abcd}} \left\{ \bar{R}_{da}(\Delta\theta_{j_D} + \Delta\theta_{j_A}) - \bar{S}_{da}(\Delta m_{j_D} + \Delta m_{j_A}) \right.$$

$$+ \bar{R}_{cd}(\Delta\theta_{k_C} + \Delta\theta_{k_D}) - \bar{S}_{cd}(\Delta/_{k_C} + \Delta m_{k_D})$$

$$- \bar{R}_{bc}(\Delta\theta_{j_B} + \Delta\theta_{j_C}) + \bar{S}_{bc}(\Delta m_{j_B} + \Delta m_{j_C})$$

$$\left. - \bar{R}_{ab}(\Delta\theta_{k_A} + \Delta\theta_{k_B}) + \bar{S}_{ab}(\Delta m_{k_A} + \Delta m_{k_B}) \right\}. \qquad (3.8)$$

In order to evaluate the values of $\bar{R}$ and $\bar{S}$, secondary control volumes are drawn about the midpoints of the edges of the primary control volume, as shown in Fig. 3.3. Considering the "east" secondary control volume in particular (Fig. 3.4), the line integration for a first derivative term becomes (using $\frac{\partial u}{\partial\theta}$ as an example,
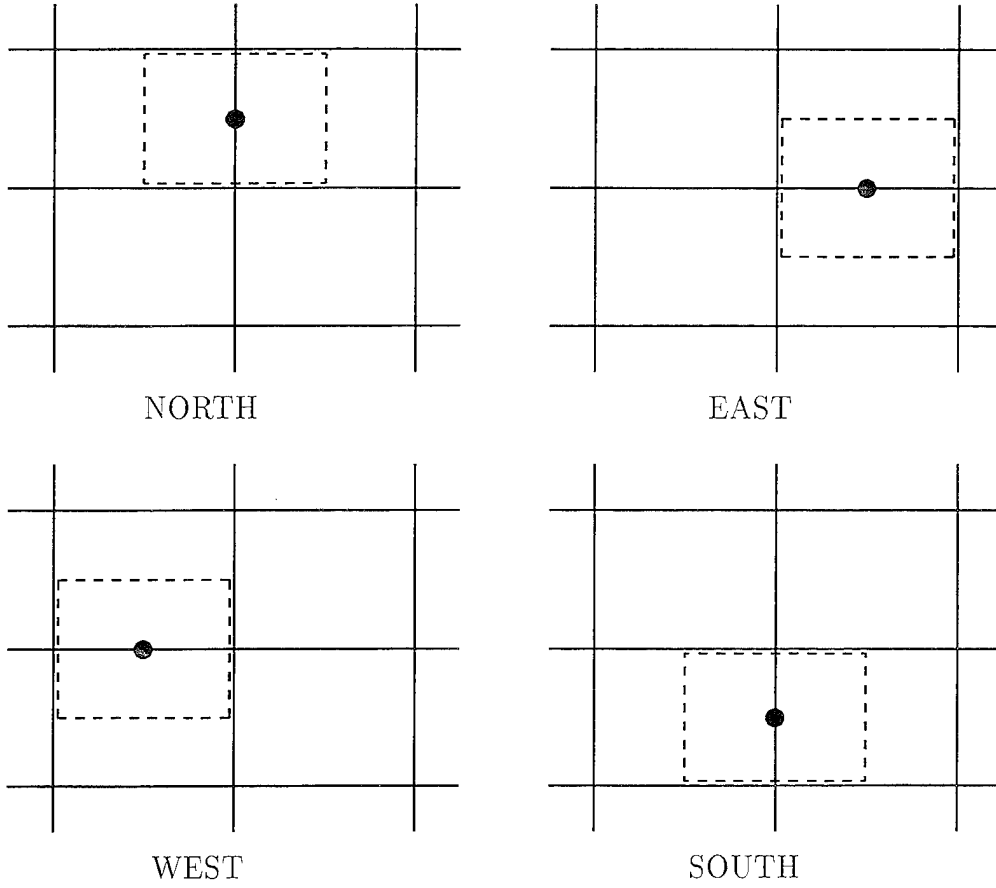
Figure 3.3: Secondary cells for viscous term integration.

where u is a representative velocity component):

$$
\begin{aligned}
\left.\frac{\partial u}{\partial \theta}\right|_{cd} &= \frac{1}{A_{add'c'bc}} \oint_{add'c'bc} (u\,dm) \\
&= \frac{1}{A_{add'c'bc}} \left\{ u_d(m_{d'} - m_{ad}) + u_{o'}(m_{c'} - m_{d'}) \right. \\
&\qquad\qquad \left. + u_c(m_{bc} - m_{c'}) + u_o(m_{ad} - m_{bc}) \right\}.
\end{aligned}
\tag{3.9}
$$

If the assumption is made that $A_C \approx A_D \approx A_{add'c'bc}$, then with the definitions given in Eqn. 3.7, Eqn. 3.9 may be split into two pieces that only contain operations

30
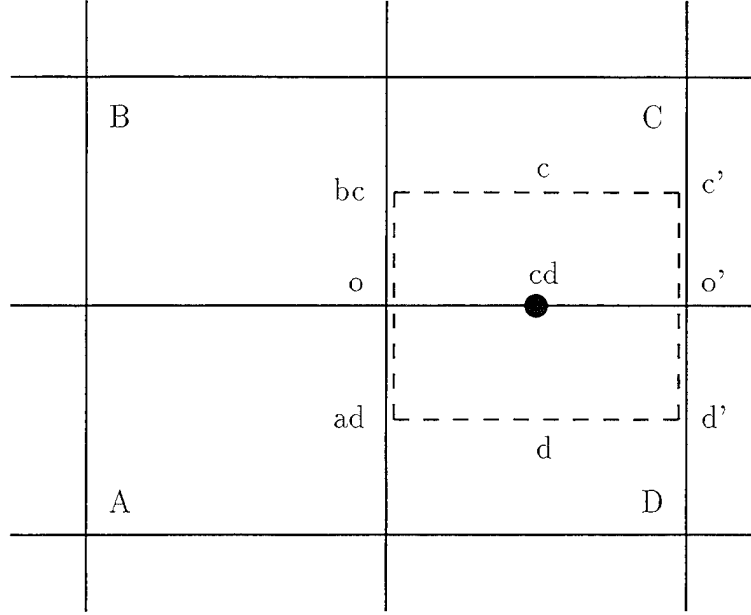
Figure 3.4: East secondary cell.

in cells C and D, respectively:

$$
\begin{aligned}
\left.\frac{\partial u}{\partial \theta}\right|_{cd} = & \ \frac{1}{A_C}\left\{u_{o'}(m_{c'} - m_{o'}) - u_c\Delta m_{j_C} + u_o(m_o - m_{bc})\right\} \\
+ & \ \frac{1}{A_D}\left\{u_{o'}(m_{o'} - m_{d'}) + u_d\Delta m_{j_D} + u_o(m_{ad} - m_o)\right\}.
\end{aligned} \tag{3.10}
$$

Similar expressions can be written for the other derivatives at cd.

Considering only the R term portion of Eqn. 3.8, with the assumption that $R = \frac{\partial u}{\partial \theta}$ (which is part of it), further assume that:

$$
A_{abcd} \approx A_A \approx \ A_B \ \approx A_C \approx A_D,
$$

$$
\Delta t_{abcd} \approx \Delta t_A \approx \Delta t_B \approx \Delta t_C \approx \Delta t_D,
$$

$$
\Delta\theta_{j_D} \approx \Delta\theta_{j_A}, \qquad \Delta\theta_{k_D} \approx \Delta\theta_{k_A},
$$

$$
\Delta\theta_{j_B} \approx \Delta\theta_{j_C}, \qquad \Delta\theta_{k_B} \approx \Delta\theta_{k_C},
$$

31

The east secondary cell contribution to ΔU_{v_o} in Eqn. 3.8 may then be written as:

$$
\begin{aligned}
\Delta U_{v_o}\big|_{east} &= \frac{\Delta t_{abcd}}{2 A_{abcd}} \bar{R}_{cd}(\Delta\theta_{k_C} + \Delta\theta_{k_D}) \\
&= \frac{\Delta t_C}{2 A_C^2} \left[\{ u_{o'}(m_{c'} - m_{o'}) - u_c \Delta m_{j_C} + u_o(m_o - m_{bc}) \} \Delta\theta_{k_C} \right] \\
&\quad + \frac{\Delta t_D}{2 A_D^2} \left[\{ u_{o'}(m_{o'} - m_{d'}) + u_d \Delta m_{j_D} + u_o(m_{ad} - m_o) \} \Delta\theta_{k_D} \right]. \, (3.11)
\end{aligned}
$$

Similar expressions may be derived for the S term in the east secondary cell and for all terms in the remaining secondary cells.

Using the method which produced Eqn. 3.11, the complete expression for the viscous fluxes at point o may be written as a summation of four terms, one for each of the four surrounding cells. Each of these four terms will use information locally available to that cell only; no information from neighboring cells is required. Therefore, the viscous terms may be calculated by a single sweep through the cells, with the contributions from each secondary cell fragment calculated and distributed to the appropriate corner nodes.

3.3   Boundary Conditions

Boundary conditions for the solver must be applied at the inflow and outflow of the flow domain, along solid surfaces, at the periodic boundary between adjacent blade passages, and at the sliding interface between adjacent blade rows.

3.3.1   Inflow and Outflow Conditions

The boundary conditions specified at the inlet and exit of the domain are determined using a modification of the characteristic variable approach described by Giles [27, 28]. This method uses a knowledge of characteristic theory to develop the appropriate boundary conditions for the flow regime under consideration. The use of

characteristic theory allows the development of non-reflective boundary conditions. These are important in turbomachinery computations due to the close proximity of the inlet and exit boundaries to the blade surfaces.

Two different sets of boundary conditions are generally used in turbomachinery computations. Transonic compressor calculations require that the inlet relative Mach number be supersonic, while the axial Mach number remains subsonic; turbine calculations use a subsonic inlet Mach number condition. Both compressor and turbine cascade computations require a subsonic exit condition with a specification of the downstream static pressure.

A complete description of the boundary condition development and implementation may be found in Appendix B.

3.3.2 Solid Surface Conditions

On solid surfaces, the application of characteristic theory is inappropriate because the assumption of inviscid flow is no longer valid. Viscous calculations require that the surface velocity of the fluid match that of the wall boundary, the so called "no-slip" condition. While this condition serves to specify the surface velocities, the surface temperature and pressure must come from a knowledge of the physical boundary conditions, coupled with a solution of the momentum equation in a direction normal to the surface. Again, compressor and turbine cases are different. Compressor airfoils, which are generally thin and operate without internal cooling, are assumed to use an adiabatic surface condition. Turbine airfoils, on the other hand, often contain internal cooling air and are considered to operate with a near constant surface temperature. (A full thermal analysis of the coupled fluid-structure domain would produce an improved boundary condition, but is beyond the scope of the current work).

With the surface velocities and temperature known, a solution of the normal momentum equation provides the surface pressure. In the solution of this equation, there are a number of components worth discussing. The first is a viscous term that remains after assuming the no-slip condition and neglecting streamwise velocity gradients:

$$\frac{\partial p}{\partial \eta} = \frac{\mu}{Re} \frac{\partial^2 v}{\partial \eta^2}, \tag{3.12}$$

where η is the direction normal to the wall. The derivative of the velocity component normal to the wall, v, will be small and is further multiplied by the inverse of the Reynolds number. Therefore, this term is typically neglected for high Reynolds number flows.

The second term to consider is the pressure gradient resulting from wall curvature. In this case:

$$\frac{\partial p}{\partial \eta} \sim \frac{u^2}{R_c}, \tag{3.13}$$

where u is the streamwise velocity component parallel to the wall and R_c is the local wall radius of curvature. This term might become important for the small radii of typical compressor blade leading and trailing edges, but the streamwise grid density required to resolve the boundary layer in these areas will usually insure that this term is also quite small, and is therefore neglected in the current analysis.

The final term results from the movement of the wall, but only for cases where the radius of the streamsheet is also changing. In this case the normal momentum equation yields:

$$\frac{\partial p}{\partial \eta} = -\rho v_\theta^2 \sin(\alpha_w) \frac{r_m}{r}, \tag{3.14}$$

where α_w is the wall angle measured from the streamwise direction. For a stationary wall, $v_\theta = 0$, and the usual condition of $\partial p / \partial \eta = 0$ is obtained. However for a moving wall, $v_\theta = \Omega r$, and this term can no longer be ignored. Experience has shown that,

34
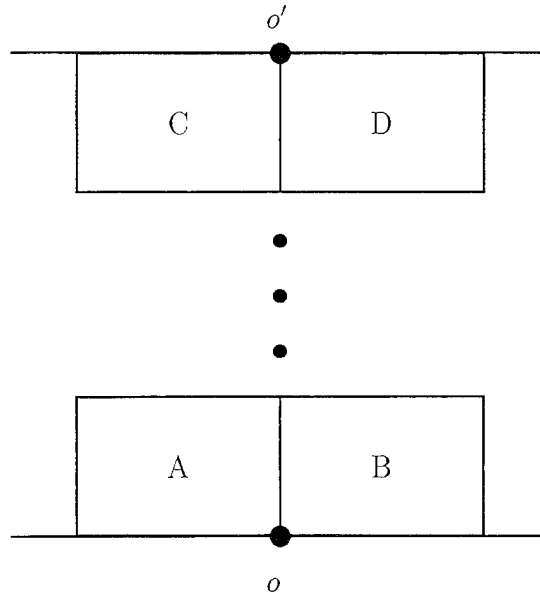
Figure 3.5: Treatment of periodicity condition.

for a typical compressor rotor computation, this term results in a 1-2% increase in the surface pressure from the $\partial p / \partial \eta = 0$ condition.

3.3.3 Periodicity

In order to simulate an infinitely repeating cascade, a periodic boundary condition is implemented upstream of the leading edge and downstream of the trailing edge of the blade. Nodes that lie along these boundaries are treated as internal points in the integration, but an extra pass is required to combine fluxes across the periodic boundary and equate the values at both points. Considering Fig. 3.5, the fluxes calculated in cells A and B during the normal flux calculation pass are sent to node o, those in cells C and D are sent to o'. The second pass adds the partial fluxes at nodes o and o' and sets both nodal values to this correct total flux. This double pass is performed during both inviscid and viscous flux evaluations, as well as during the calculation of the artificial dissipation terms.

35

For cases which contain more than one passage in a blade row, the periodicity condition is still applied, but only along the top and bottom boundaries of the multiple passage grid block. The grid lines upstream and downstream of the blades in the middle of the multiple passage block are implemented as internal points, and no special treatment is required.

3.3.4 Intra-Blade Row Interface Conditions

The interface between adjacent blade rows occurs along an overlap region of two grid lines, where the overlapping lines lie directly on top of one another. Requiring the grid lines to slide on top of one another significantly simplifies the interpolation procedure, which may be implemented as a series of line interpolations rather than a more expensive area interpolation. The speed at which the interpolation is done is offset by the fact that it must be done twice in each overlap region of a blade row for each stage of the n-stage Runge-Kutta time integration procedure. Since all blade rows in the calculation are computed in the absolute reference frame, there is no need to impose velocity triangle relationships at the row interfaces.

The treatment of the flux variables at the interface is done in a conservative manner similar to that of Rai [52, 53]. The procedure simply requires an integration of the flow variables before and after the interpolation between grids. The interpolated values are then adjusted by the ratio of the two integrated values. This insures that both the distribution of the variable, as well as its integrated value, are preserved across the interface boundary.

Choice of the interpolation algorithm is important, particularly when the grid is relatively coarse. Not only is it important to conserve the flow variables across the interface, but it is also necessary to pass their profiles with as little distortion as possible. To this end, three different interpolation functions were implemented, each based on Lagrangian polynomials, differing only in the number of points used

in the interpolation. In general, the value of a function at some point $f(x)$ may be found from values at surrounding points, $f(x_0), f(x_1), \ldots f(x_N)$, by using the following formula [14, pg. 27]:

$$f(x) = \sum_{i=0}^{N} \left[\prod_{\substack{j=0 \\ j \neq i}}^{N} \frac{(x - x_j)}{(x_i - x_j)} \right] f(x_i) = \sum_{i=0}^{N} L_i(x) f(x_i), \tag{3.15}$$

where $L_i(x)$ are the Lagrange polynomials. For example, a three point interpolation is written as:

$$f(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} f(x_0) + \frac{(x - x_2)(x - x_0)}{(x_1 - x_2)(x_1 - x_0)} f(x_1)$$
$$+ \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} f(x_2). \tag{3.16}$$

This method of interpolation uses a polynomial of degree N to locally fit the existing data and determine the value of the desired point; i.e., $N = 1$ uses a linear fit, $N = 2$ uses a quadratic function, etc. Interpolation has been implemented with $N = 1, 2,$ and 3 corresponding to two, three, and four points respectively. In each case, the points used in the interpolation are found by requiring that the desired point lie between the two center points; the third point in a three-point interpolation is located at the closest of the next two outer points. The determination of the Lagrange polynomials is made once each time the grid position is indexed; these values are stored and reused for each interpolation performed at this grid position.

An example of the value of the higher-order interpolation may be found in the density profile from the hot-streak calculations described in Sect. 6.1.4. Starting with identical profiles, an interpolation is performed using each of the Lagrangian schemes; the resulting profiles are shown in Fig. 3.6. Not surprisingly, the three-point algorithm provides a significantly better profile near the center of the profile while

all methods are indistinguishable in the linear regions. The four- and three-point interpolations are nearly identical, as expected, because of the quadratic nature of the profile. Since the flow equations are nonlinear, and are at most of quadratic order, the three-point function should be sufficient to predict most smooth profiles, and in practice requires little computational overhead beyond that of the two-point function.

3.4   Artificial Dissipation

The explicit addition of artificial dissipation is generally required for time-marching schemes with spatial central differencing. Damping is needed both to suppress spurious oscillations in smooth regions of the flow, as well as to capture flow discontinuities, such as shocks. This section examines the method in which the second- and fourth-order damping terms are added to the code.

Second-order smoothing is required by central-difference schemes in order to capture discontinuities in the flow, such as shock waves. While a shock would ideally have zero thickness, central difference schemes do not permit this type of discontinuity as a valid solution. The introduction of second-order smoothing serves to locally change the difference equations to first-order, which do permit discontinuities. Shocks are smeared over several grid points with a minimum amount of oscillation (undershoot and overshoot) up- and downstream of the shock.

Fourth-order smoothing is applied to suppress the formation of the odd-even, or "sawtooth," modes characteristic of central difference schemes. A central-difference scheme uses a three-point stencil in order to evaluate first derivatives, but does not use the value at the center node. This causes an uncoupling of values at alternate grid points, resulting in the formation of the so called odd-even modes shown in Fig. 3.7. Using a three-point stencil to calculate the first derivative $\frac{\partial f}{\partial x}$ of the solution in this
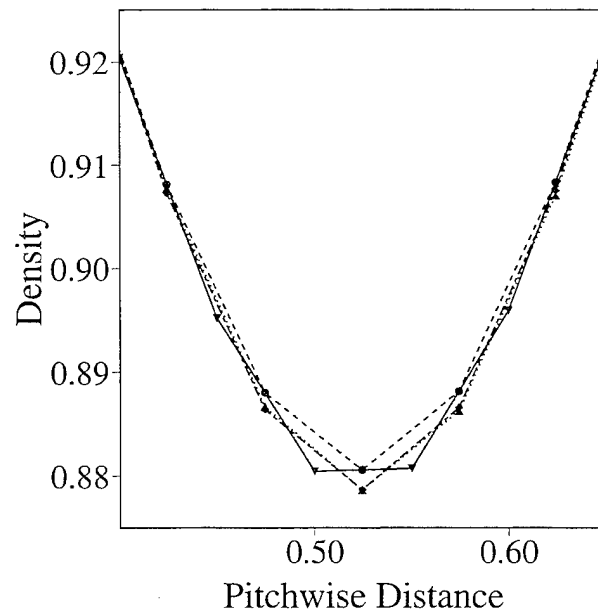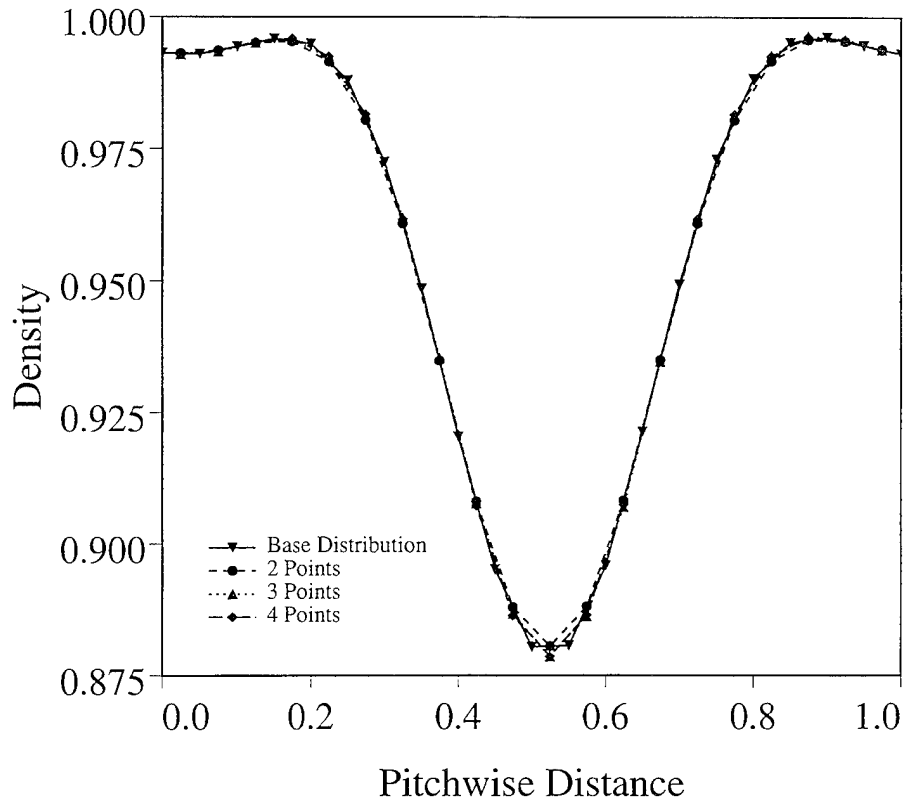
Figure 3.6: Comparison of interpolation polynomials in hot streak calculation.
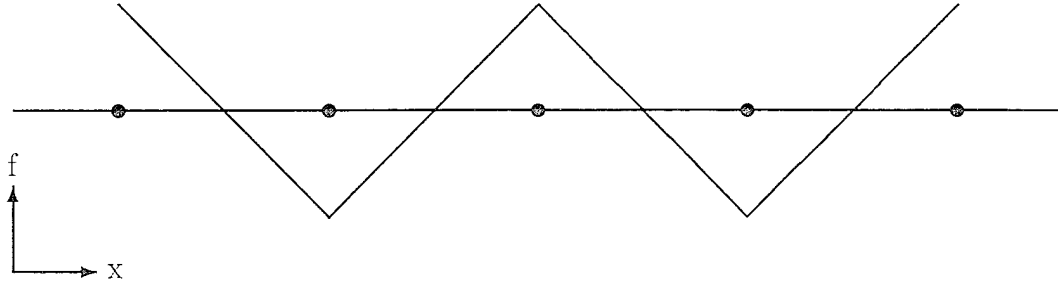
Figure 3.7: Odd-even or sawtooth mode.

figure would yield a value of zero at all points. Therefore, this solution would be accepted as a valid numerical solution of $\frac{\partial f}{\partial x} = 0$, while being wholly unsatisfactory. Fourth-order damping uses a five-point computational stencil to detect and suppress these oscillations.

The smoothing used in the current effort is an unstructured implementation of the damping terms normally found in finite difference solutions. The smoothing terms are explicitly added to the right hand side of Eqn. 2.15. The combined second- and fourth-order smoothing operator has the form:

$$D\left(U\right) = D_j\left(U\right) + D_k\left(U\right),\qquad(3.17)$$

where again j and k are the cell local grid directions. In the j-direction, the operator is given by:

$$D_j\left(U\right) = \beta_j\left(\epsilon_2\delta_j^2 U - \epsilon_4\delta_j^4 U\right),\qquad(3.18)$$

where the δ_j^2 and δ_j^4 operators represent the second and fourth differences in the j-direction. A similar expression may be written for the k-direction. In practice, a sweep is made through the computational domain to calculate the second differences

40

at all nodes. The fourth difference is then found with another sweep to "second difference the second differences." The coefficients, β_j and β_k are approximations of the cell spectral radius, scaling the added dissipation to the change in the flux residual. One approach is use the β values to cancel the $\Delta t/A$ term which results from the integration of Eqn. 2.15. In this case, the β's have the form [36, 39]:

$$\beta_j = \beta_k = \frac{A}{\Delta t^*}. \tag{3.19}$$

Kroll et al. [39] define Δt^* as the local cell time step based on an unit Courant-Freidrichs-Lewy (CFL) number, where the CFL number is defined as the ratio of the time step of the numerical scheme to the transit time across a characteristic cell dimension at the local maximum wavespeed [17]. An equivalent form is given by Jorgenson and Chima [36], who use the local time step calculated as a function of the stability limit of the cell, but use a subsequent increase in the second- and fourth-order damping coefficients described below. An alternate form is given by Mavriplis and Jameson [47], who scale the dissipation depending on the stretching in the local grid coordinates. A modified form of this latter approach is used in the current effort, with:

$$\beta_j = \lambda_j \left(1 + \left(\frac{\lambda_k}{\lambda_j} \right)^{\frac{2}{3}} \right),$$
$$\beta_k = \lambda_k \left(1 + \left(\frac{\lambda_j}{\lambda_k} \right)^{\frac{2}{3}} \right). \tag{3.20}$$

Here, λ_j and λ_k are the maximum eigenvalues in the two local grid directions:

$$\lambda_j = \left| v_m \Delta\theta_j - w_\theta \frac{\Delta m_j}{r} \right| + c \sqrt{\Delta\theta_j^2 + \left(\frac{\Delta m_k}{r} \right)^2},$$
$$\lambda_k = \left| v_m \Delta\theta_k - w_\theta \frac{\Delta m_k}{r} \right| + c \sqrt{\Delta\theta_k^2 + \left(\frac{\Delta m_k}{r} \right)^2}, \tag{3.21}$$

where c is the local speed of sound, and $\Delta\theta_j$, Δm_j, etc. are given by Eqn. 3.7.

The ϵ_2 and ϵ_4 coefficients are given by:

$$\epsilon_2 = \sigma_2 \Delta P, \tag{3.22}$$

$$\epsilon_4 = \max(0, \sigma_4 - \epsilon_2). \tag{3.23}$$

The pressure switch ΔP is implemented as a normalized second difference in pressure at a particular node:

$$\Delta P_i = \frac{\left|\sum_{i \neq n}(p_i - p_n)\right|}{\left|\sum_{i \neq n}(p_i + p_n)\right|}, \tag{3.24}$$

where the n points are the edge nodes around the four cells bordering a particular node. Near a shock wave, where the fourth-order smoothing becomes destabilizing, the pressure gradient, and therefore ΔP, becomes large. This causes the $\sigma_2 \Delta P$ term to dominate σ_4 in Eqn. 3.23, switching off the fourth-order smoothing. In smooth regions of the flow, ΔP is small. The dissipative terms are of third order and do not reduce the accuracy of the central-difference scheme. Near shock waves, the pressure switch becomes large and the dissipation becomes locally of first order. The coefficients σ_2 and σ_4 are:

$$\sigma_2 = \mathcal{O}(1),$$
$$\sigma_4 = \mathcal{O}(\frac{1}{16}). \tag{3.25}$$

The smoothing terms are further modified by using a velocity scaling in order to avoid contamination of the viscous terms in the boundary layer. Both smoothing terms are multiplied by a factor which scales with the ratio of the local velocity to that determined from the local static-to-total pressure ratio [21]. For rotational flows, care must be taken to insure that this ratio is formed with the local *relative*

velocity and total pressure to insure proper detection of the boundary layer on a rotating blade.

3.5 Temporal Integration Scheme

The following sections describe the numerical integration scheme used to advance the flow equations forward in time. The first section describes the dual time stepping algorithm, which allows an explicit multi-stage time-stepping method to be used as a driver for a fully implicit scheme. A description is then presented of the two convergence acceleration techniques implemented in the current effort.

3.5.1 Dual Time Stepping

While explicit algorithms, and particularly multi-stage Runge-Kutta schemes, have a low operation count per iteration, they suffer a serious time step restriction as compared to their implicit counterparts. This condition is further exacerbated in unsteady flows where the global minimum time step must be used throughout the domain. In these cases, there may be a variation of several orders-of-magnitude in the time step required for stability between the largest and smallest cells. Additionally, a direct temporal integration of Eqn. 2.15 prohibits the use of convergence acceleration techniques, such as local time stepping, implicit residual smoothing, or multigrid, for unsteady flows.

To overcome these limitations, the Navier-Stokes equations may be reformulated using one of the dual time step methods described by Jameson [32] and Weiss and Smith [76]. These methods use a standard explicit scheme as the driver for a fully implicit time stepping scheme, differing only in the details of their implementation (which will be discussed in Section 3.5.3 below). Updates of the solution domain

43

are performed at time intervals determined by the required temporal accuracy of the solution, rather than those prescribed by the smallest cells in the domain.

A compact, differential form of Eqn. 2.15 may be written as:

$$A\frac{\partial U}{\partial t} + \mathcal{R}(U) = 0, \qquad (3.26)$$

where $\mathcal{R}(U)$ is the residual containing both the convective and diffusive fluxes, as well as the artificial dissipation terms. In a steady flow case, both terms in a discretized form of Eqn. 3.26 tend toward zero as the solution converges. Acceleration methods are effective because they drive these terms to zero with fewer iterations, leaving no residual effect on the solution. However, transients computed using these acceleration methods are completely non-physical since time has not been advanced uniformly across the mesh. Therefore, when using these techniques, the variable t in Eqn. 3.26 must be considered solely as an integration variable and not as physical time. This point will become important in the construction of the dual time step formulation.

In an unsteady flow case, neither term in Eqn. 3.26 is, in general, zero, yet the sum of the two terms is still zero for all time. For temporal accuracy, time must be advanced uniformly throughout the mesh, disallowing the use of any form of acceleration technique. In the construction of a dual time step method, a new residual is defined which includes both terms in Eqn. 3.26. This combined residual is then driven to zero using a multi-stage Runge-Kutta algorithm. Recasting the equations in this way produces a formulation analogous to that used in the solution of the steady-state equations, and again allows the use of acceleration methods.

A fully implicit formulation of Eqn. 3.26 may be approximated as:

$$AD_t(U^{(n+1)}) + \mathcal{R}(U^{(n+1)}) = 0. \qquad (3.27)$$

44

where the superscript denotes the time level and $D_t(U)$ is a backwards-difference temporal operator of some specified accuracy. Using a second-order backwards-difference operator, Eqn. 3.27 becomes:

$$\frac{A}{2\Delta t}\left[3U^{(n+1)} - 4U^{(n)} + U^{(n-1)}\right] + \mathcal{R}(U^{(n+1)}) = 0, \qquad (3.28)$$

which is the combined residual previously described. Because this combined residual will be driven to zero in a converged solution, Eqn. 3.28 may be treated as a modified steady-state problem, solved by stepping in a fictitious time τ:

$$-A\frac{\partial U^{(n+1)}}{\partial \tau} = \frac{A}{2\Delta t}\left[3U^{(n+1)} - 4U^{(n)} + U^{(n-1)}\right] + \mathcal{R}(U^{(n+1)}) = \mathcal{R}^*(U^{(n+1)}) \quad (3.29)$$

or

$$A\frac{\partial U^{(n+1)}}{\partial \tau} + \mathcal{R}^*(U^{(n+1)}) = 0, \qquad (3.30)$$

which is of the same form as Eqn. 3.26. Between each increment in real time t, the solution converges in the fictitious time variable τ, causing the $\partial U^{(n+1)}/\partial \tau$ term to vanish, resulting in a proper solution of Eqn. 3.28. Selection of the physical time increment Δt is now driven by the desired time accuracy of the solution, rather than by the global minimum time step restriction. The progression of the solution in fictitious time does not affect the solution in real time, therefore any of the conventional convergence acceleration techniques may be used. In the current implementation, both local time stepping and implicit residual smoothing (described in the following section) have been implemented.

Incorporating the dual time stepping into a standard Runge-Kutta code requires little additional effort. The backward-difference operator of Eqn. 3.29 is included as an additional source term. This source term and the storage of the solution at the previous time level, are the only modifications required. The complete

algorithm for the dual-time iterations is outlined in Fig. 3.8, where the number of iterations in real-time is specified with the $nreal$ parameter and $nfake$ is the number of pseudo-time iterations required to converge to the solution of $U^{(n+1)}$.

Initialization of the value of $U^{(n+1)}$ at the beginning of a sequence of fictitious time steps is performed by predicting its value using the previous value of the time derivative. In practice, it has been found necessary to limit the change in the value of a dependent variable to approximately 30% of its current value to prevent destabilizing oscillations of flow variables which have values near zero.

Use of the dual time step integration scheme requires the specification of a suitable real time step which is directly related to the desired temporal accuracy of the solution. For periodic solutions, the time step is specified by a division of the period of the solution into a number of equal increments; non-periodic solutions require a direct specification of the real time step value. The choice of the real time step is a topic to be discussed in Chapter 6.

3.5.2 Convergence Acceleration

Two techniques for accelerating the convergence of the numerical scheme: local time stepping and implicit residual smoothing. Both of these methods are applicable to flow cases converging to a steady-state or those using the dual time stepping described in the previous section.

Local time stepping allows each cell to use its maximum permissible time step based on a local stability analysis. Viscous flow cases require that both the convective and diffusive terms be included in this analysis. Assuming the use of the dual time stepping algorithm, the local pseudo-time step is given by:

$$\Delta \tau = CFL \left(\frac{\Delta \tau_i \, \Delta \tau_v}{\Delta \tau_i + \Delta \tau_v} \right), \tag{3.31}$$

<u>**initialize**</u>

$U^{(n)} = U$ from initial conditions

$U^{(n-1)} = U^{(n)}$

$\dfrac{\partial U}{\partial t} = 0$

<u>**loop in real-time**</u>

do nr = 1, nreal

$$U^{(n+1)}_{old} = U^n + \Delta t \dfrac{\partial U}{\partial t}$$

<u>**loop in pseudo-time**</u>

do nf = 1, nfake

do ns = 1, nstage

Runge-Kutta update of $U^{(n+1)}_{new}$ using $U^{(n+1)}_{old}$ and $\dfrac{\partial U}{\partial t}$

$$\dfrac{\partial U}{\partial t} = \dfrac{3U^{(n+1)}_{new} - 2U^{(n)} + U^{(n-1)}}{2\Delta t}$$

end do

$U^{(n+1)}_{old} = U^{(n+1)}_{new}$

end do

$U^{(n-1)} = U^{(n)}$

$U^{(n)} = U^{(n+1)}_{old}$

end do

Figure 3.8: Dual time stepping algorithm.

where CFL is the allowable Courant-Friedrichs-Lewy stability limit of the particular time integration scheme, and $\Delta\tau_i$ and $\Delta\tau_v$ are the local convective and diffusive time step limits, respectively. The convective limit used here is similar to that described by Jorgenson and Chima [36] for calculation of quasi-3D flows, with modifications into the appropriate finite volume form:

$$\Delta\tau_i = \frac{A}{\left[\mid v_m \mid \Delta_j + \mid w_\theta \mid \Delta_k + c\sqrt{\Delta_j^2 + \Delta_k^2} + \dfrac{v_\theta dr/dmA^2}{2c\sqrt{\Delta_j^2 + \Delta_k^2}}\right]},\tag{3.32}$$

where

$$\Delta_j = \frac{\mid \Delta m_j \mid}{r} + \mid \Delta\theta_j \mid$$
$$\Delta_k = \frac{\mid \Delta m_k \mid}{r} + \mid \Delta\theta_k \mid,$$

and c is the local sound speed. The viscous limit is a finite volume formulation of the diffusive limit given by Arnone, et al.[4]:

$$\Delta\tau_v = \frac{A^2}{\dfrac{2.5\gamma}{\rho}\left(\dfrac{\mu}{Pr} + \dfrac{\mu_T}{Pr_T}\right)\left(\Delta_j^2 + \Delta_k^2\right)}.\tag{3.33}$$

All quantities are assumed to be cell-averaged values.

Implicit residual smoothing (IRS) is used to remove spikes from the flux residuals, reducing the high-frequency errors. Use of IRS can expand the stability region and improve the damping characteristics of multi-stage time stepping procedures. For two-dimensional problems, the total flux residual $\mathcal{R}^*(U)$ of Eqn. 3.29 is replaced by a solution of the implicit equations [47]:

$$\overline{\mathcal{R}_i^*} = \mathcal{R}_i^* + \epsilon\nabla^2\overline{\mathcal{R}_i^*},\tag{3.34}$$

where i is the node point index, $\nabla^2\overline{\mathcal{R}_i^*}$ is the undivided Laplacian of the residual, and ϵ is the smoothing coefficient. While the smoothing can be applied using a uniform value of ϵ, coefficients which are functions of the local grid stretching and flow conditions have been shown to greatly improve the rate of convergence [51]. The use of locally varying coefficients makes the scheme more implicit in the grid direction of higher stretching, and less implicit in the direction with less stretching. In an unstructured quadrilateral mesh, the Laplacian operator of Eqn. 3.34 may be replaced by:

$$\overline{\mathcal{R}_i^*} = \mathcal{R}_i^* + \epsilon_j\,\delta_j^2\overline{\mathcal{R}_i^*} + \epsilon_k\,\delta_k^2\overline{\mathcal{R}_i^*}, \tag{3.35}$$

where again j and k are the characteristic directions within the cell and δ^2 is the second difference operator in the appropriate direction. The smoothing coefficients are [47]:

$$\begin{aligned}
\epsilon_j &= \max\left(\frac{1}{4}\left[\left(\frac{CFL}{CFL^*}\,\beta_j\right)^2 - 1\right],\,0\right), \\
\epsilon_k &= \max\left(\frac{1}{4}\left[\left(\frac{CFL}{CFL^*}\,\beta_k\right)^2 - 1\right],\,0\right),
\end{aligned} \tag{3.36}$$

where β_j and β_k are the scaling coefficients given by Eqn. 3.20. The $\frac{CFL}{CFL^*}$ term is the ratio of the CFL numbers in the smoothed and unsmoothed schemes and has been found to have an optimum value of $\frac{CFL}{CFL^*} \approx 2$ [72]. The second difference operators are similar to those used in the artificial dissipation and are formed in an analogous manner. Two Jacobi iterations are used to provide an approximate solution to Eqn. 3.35; the smoothing operator is applied during each stage of the Runge-Kutta procedure. In practice, the use of IRS technique reduces the number of pseudo-time iterations required to converge the solution at each real time step by approximately one-third, with only a small increase in the computational time required for each iteration.

3.5.3 Numerical Scheme

The time integration algorithm used in the present work is an n-stage Runge-Kutta method. In the classical Runge-Kutta scheme, the residuals are evaluated and stored at each stage, then combined in the last stage to update the solution. Storage of these residuals can be quite expensive and normally leads to the use of a modified Runge-Kutta scheme where a given stage's residuals are used only in the evaluation of the next stage.

The Runge-Kutta scheme used to update the solution in pseudo-time is one of the modified schemes of Jameson [32]. In general, an r-stage scheme may be given by:

$$
\begin{aligned}
U^{(n+1,0)} &= U^{(n)} \\
U^{(n+1,1)} &= U^{(n)} - \alpha_1 \frac{\Delta \tau}{A} \left[Q^{(0)} + D^{(0)} \right] \\
&\vdots \\
U^{(n+1,q)} &= U^{(n)} - \alpha_q \frac{\Delta \tau}{A} \left[Q^{(q-1)} + D^{(q-1)} \right] \\
&\vdots \\
U^{(n+1)} &= U^{(n+1,r)},
\end{aligned}
\tag{3.37}
$$

where

$$
Q^{(0)} = Q\left[U^{(n)} \right],
\tag{3.38}
$$

$$
D^{(0)} = D\left[U^{(n)} \right],
\tag{3.39}
$$

and

$$
Q^{(q)} = Q\left[U^{(n+1,q)} \right]
\tag{3.40}
$$

$$D^{(q)} = \beta_q D\left[U^{(n+1,q)}\right] + (1 - \beta_q)D^{(q-1)}. \tag{3.41}$$

Here $Q[U]$ is the combined inviscid and viscous flux residual and $D[U]$ is the artificial dissipation operator. Two modified schemes are given by Jameson [32]. The first is a four-stage scheme with two evaluations of the artificial dissipation:

$$
\begin{aligned}
\alpha_1 &= \frac{1}{3}, & \beta_1 &= 1 \\
\alpha_2 &= \frac{4}{15}, & \beta_2 &= \frac{1}{2} \\
\alpha_3 &= \frac{5}{9}, & \beta_3 &= 0 \\
\alpha_4 &= 1, & \beta_4 &= 0.
\end{aligned}
\tag{3.42}
$$

The second is a five-stage scheme with three dissipation evaluations:

$$
\begin{aligned}
\alpha_1 &= \frac{1}{4}, & \beta_1 &= 1 \\
\alpha_2 &= \frac{1}{6}, & \beta_2 &= 0 \\
\alpha_3 &= \frac{3}{8}, & \beta_3 &= 0.56 \\
\alpha_4 &= \frac{1}{2}, & \beta_4 &= 0 \\
\alpha_5 &= 1, & \beta_5 &= 0.44.
\end{aligned}
\tag{3.43}
$$

Both of these schemes have first order temporal accuracy (in pseudo-time) and yield a maximum $CFL = 3.0$ for the four-stage scheme, and $CFL = 4.0$ for the five-stage scheme. These values are typically reduced to 2.8 and 3.5 respectively with the inclusion of added dissipation [36, 39]. Both of these schemes have been used in the current effort: the five-stage is commonly used for the flow equations and for the one-equation turbulence model on coarse grids. Adapted meshes often

require the use of sub-iterations of the four-stage scheme for the integration of the one-equation turbulence model (discussed in Chapter 4).

In Jameson's implementation of the dual time stepping method [32], the Runge-Kutta scheme to solve Eqn. 3.30 in the form as described above. The temporal derivative source term is evaluated with the $U^{(n+1)}$ from the previous stage. The evaluation of the q-th stage is given by:

$$
\begin{aligned}
U^{(n+1,q)} &= U^{(n)} - \alpha_q \frac{\Delta\tau}{A} \left(Q^{(q-1)} + D^{(q-1)} \right. \\
&\left. - \frac{A}{2\Delta t} \left[3U^{(n+1,q-1)} - 4U^{(n)} + U^{(n-1)} \right] \right).
\end{aligned}
\tag{3.44}
$$

Arnone, et al.[4] have noted that, although this time discretization of Eqn. 3.29 is implicit, stability problems can occur in Jameson's method if the time step in pseudo-time exceeds the physical time step. Therefore, the time step τ must be limited [4]:

$$
\Delta\tau = \min \left\{ \Delta\tau, \frac{\Delta t}{\frac{3}{2} \frac{CFL}{CFL^*}} \right\},
\tag{3.45}
$$

for solutions without multigrid. The effect of this limitation normally only appears in viscous calculations, where the time steps in the inviscid core region of the flow can often be one to two orders-of-magnitude higher than those in the viscous layer.

An alternate implementation of dual time stepping is given by Weiss and Smith [76] who uses a partially implicit treatment of the temporal derivative term. In this case, the q-th stage of the Runge-Kutta scheme becomes:

$$
\begin{aligned}
\left(I + \frac{3\Delta\tau}{2\Delta t} \right) \Delta U &= -\alpha_q \frac{\Delta\tau}{A} \left(Q^{(q-1)} + D^{(q-1)} \right. \\
&\left. - \frac{A}{2\Delta t} \left[3U^{(n+1,q-1)} - 4U^{(n)} + U^{(n-1)} \right] \right),
\end{aligned}
\tag{3.46}
$$

where $\Delta U = U^{(n+1,q)} - U^{(n)}$. This scheme does not require the pseudo-time step limitation described above.

An alternate scheme has been used for the calculations presented in this thesis. This method is a simple reformulation of Eqn. 3.44 with a fully implicit treatment of the temporal derivative:

$$
\begin{aligned}
\left(I + \frac{3\alpha_q \Delta\tau}{2\Delta t}\right) U^{(n+1,q)} &= U^{(n)} - \alpha_q \frac{\Delta\tau}{A} \left(Q^{(q-1)} + D^{(q-1)}\right. \\
&\left. + \frac{A}{2\Delta t}\left[4U^{(n)} - U^{(n-1)}\right]\right).
\end{aligned}
\tag{3.47}
$$

Like the scheme of Weiss and Smith, this scheme has not exhibited the instabilities which require the limitation on the pseudo-time step.

While all of these schemes have been implemented in the computer code developed for this thesis, studies performed to determine the adequate level of temporal resolution produced spurious behavior with Jameson's method. In particular, solutions with increasingly smaller real time steps should asymptotically approach the solution obtained using a standard explicit formulation. In some cases, however, refining the time step produced an increasing level of divergence of the dual time step solution. This phenomena remains unexplained; it has never been observed when using either the formulation of Weiss and Smith or that in Eqn. 3.47.

3.6 Properties of the Unstructured Scheme

The unstructured time integration scheme is now examined with respect to its stability, accuracy, and conservation.

3.6.1 Stability

In order to examine the stability properties of the multi-stage Runge-Kutta time integration scheme, the following model problem is considered:

$$u_t + u_x + \mu \Delta x^3 u_{xxxx} = 0, \tag{3.48}$$

where the $\mu \Delta x^3 u_{xxxx}$ term represents an added third-order dissipation. With $\mu = 0$, this equation represents the standard one-dimensional wave equation with unit characteristic speed. If the spatial derivatives are approximated with central differences, then the following system of first-order differential equations is obtained:

$$\frac{d}{dt} u_j + P_j = 0, \tag{3.49}$$

where

$$
\begin{aligned}
\Delta t \cdot P_j &= \frac{\lambda \Delta t}{2}(u_{j+1} - u_{j-1}) \\
&- \mu \frac{\lambda \Delta t}{2}(u_{j+2} - 4u_{j+1} + 6u_j - 4u_{j-1} + u_{j-2})
\end{aligned}
\tag{3.50}
$$

and $\lambda = \Delta t / \Delta x$ is the one-dimensional CFL number.

Considering the von Neumann stability analysis presented in [39], the solution to Eqn. 3.50 may be represented by the summation of a series of Fourier modes expanded in the x-direction:

$$u = \sum_k \hat{U}_k e^{ip_k(k\Delta x)}, \tag{3.51}$$

where $\hat{U}_k$ is the amplitude, p_k is the wave number of the k-th mode, and the number of terms k is equal to the number of grid points in the mesh. As this is a linear

equation, the effect of a single Fourier mode may be considered and the final solution obtained through linear supposition. Substituting a single mode into Eqn. 3.49, yields:

$$\Delta t \frac{d}{dt} \left(\hat{U} e^{ip(j\Delta x)} \right) = -\Delta t \cdot P \left(\hat{U} e^{ip(j\Delta x)} \right)$$
$$= z(\xi) \hat{U} e^{ip(j\Delta x)}, \tag{3.52}$$

where $\xi = p\Delta x$ is the phase angle and $z(\xi)$ is the Fourier symbol of the spatial discretization given by:

$$z(\xi) = -\lambda \left[4\mu(1 - \cos \xi)^2 + i \sin \xi \right]. \tag{3.53}$$

The amplification factor of the time integration scheme is then defined as:

$$g(z) = \frac{\hat{U}^{n+1}}{\hat{U}^n}, \tag{3.54}$$

and the stability region is defined as the region in the complex plane where $|g(z)| \leq 1$. For the scheme to be stable, $z(\xi)$ must lie within this stability region for all $-\pi \leq \xi \leq \pi$. A thorough analysis and description of the stability boundaries for various Runge-Kutta schemes is presented in [39].

3.6.2 Accuracy

The basic central difference scheme has second-order spatial accuracy on uniform meshes [37]. However, grid non-uniformities, particularly grid stretching and grid skew, can lower the local accuracy to first-order. In both the inviscid and viscous portions of the flow domain, the integration of flow variables to form either first or second derivatives is performed around the boundaries of some computational

cell. It is assumed in this integration process that the center of the cell lies at the node point at which the quantity is desired. Green's function is used to evaluate these derivatives, yielding a cell-average derivative value at the true centroid of the cell. Grid stretching reduces the accuracy by shifting the centroid of the cell away from the corresponding node point. This may be seen by performing a Taylor's series expansion of the first derivative about some point (x_o, y_o). Expanding Green's function:

$$
\begin{aligned}
\frac{\partial u}{\partial x}(x, y) &= \frac{1}{A} \int \int_A \frac{\partial u}{\partial x}(x_o, y_o) dA \\
&+ \frac{1}{A} \int \int_A (x - x_o) \frac{\partial^2 u}{\partial^2 x}(x_o, y_o) dA \\
&+ \frac{1}{A} \int \int_A (y - y_o) \frac{\partial^2 u}{\partial^2 x}(x_o, y_o) dA + \cdots.
\end{aligned}
\tag{3.55}
$$

For all of the higher order terms to cancel we must have

$$
\int \int_A (x - x_o)^m (y - y_o)^n dA = 0
\tag{3.56}
$$

for all $m, n = 0, 1, 2, \ldots$, implying that (x, y) must lie at (x_o, y_o). In other words, the use of Green's function, when the true cell center does not correspond to the assumed nodal point center, will introduce the higher-order error terms of Eqn. 3.55. This shift in the node point from the true cell center arises through stretching in the grid as shown in Fig. 3.9. The viscous portion of the solver is particularly sensitive to grid stretching errors due to the use of the secondary cells. In practice, a maximum stretching factor of 1.2 has been found to yield acceptable results in regions where viscous terms are important, with considerably larger stretching allowed in the inviscid portion of the domain.
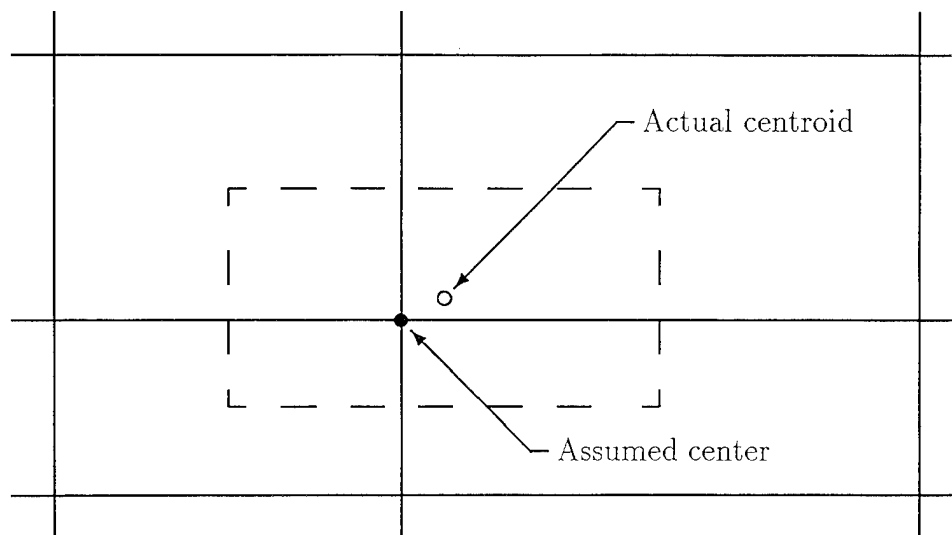
Figure 3.9: Shift of cell centroid through grid stretching.

As will be discussed in Chapter 5, mesh adaptation can have an additional impact on solution accuracy, but careful selection of the divided cell geometry can aid in an overall reduction of grid stretching errors.

3.6.3 Conservation

In addition to the consideration of a scheme's accuracy, consistency, and stability, one must also insure that the dependent variables are globally conserved within the computational domain. Demonstrating this will show a scheme's *conservation property*. The Navier-Stokes equations express conservation of mass, momentum, and energy. To show global conservation in a discrete solution of these equations, the equations must satisfy an equivalent discrete equation that approximates the integral equation. The summation of changes to the dependent variables within the given flow domain must then be equal only to the changes of the boundary nodes and the overall contribution of any internal source terms.

Considering the integration of the inviscid terms over the domain:

$$\frac{d}{dt} \int\int_{domain} U_i \, dm \, d\theta + \oint_{\substack{domain \\ boundary}} (F \, d\theta - G \, dm) = \int\int_{domain} H_i \, dm \, d\theta. \qquad (3.57)$$

Changes in the dependent variables calculated in each cell are distributed equally to each of the corner nodes in a cell. Therefore, the changes in the nodal values over the domain can be expressed as a summation of the changes within the cell and the boundary and source terms:

$$\sum_{nodes} (\delta U_i) = \sum_{cells} \left(\sum_{\substack{cell \\ nodes}} (\delta U_i) \right) + \text{boundary and source terms.} \qquad (3.58)$$

The inner summation on the right hand side is simply the change in the dependent variables within a single cell:

$$\sum_{\substack{cell \\ nodes}} (\delta U_i) = \Delta U_i. \qquad (3.59)$$

Considering a single cell, as in Fig. 3.10, a summation of the boundary fluxes, plus the inclusion of the average value of the source term within the cell, comprises the ΔU_i term:

$$\Delta U_i = -\frac{\Delta t}{A} \left[\mathcal{F}_w - \mathcal{F}_e + \mathcal{F}_s - \mathcal{F}_n \right] + \Delta t \bar{H}_i, \qquad (3.60)$$

where $\mathcal{F}$ is the flux through the boundary denoted by the appropriate subscript. For cells that share a common face, opposite signs on ingoing and outgoing fluxes yield a cancellation of fluxes along this shared face, except for cells with faces along domain boundaries. Assuming that we have a constant $\Delta t / A$ ratio throughout the
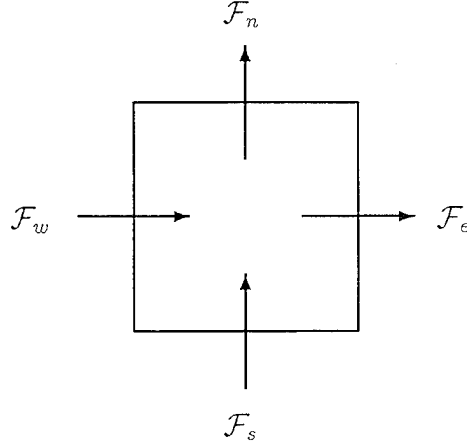
Figure 3.10: Cell flux notation.

domain:

$$\sum_{nodes} (\delta U_i) = \frac{\Delta t}{A} \left\{ \sum_{\substack{domain \\ boundary}} \mathcal{F} + \sum_{nodes} \bar{H}_i A \right\} + \text{boundary terms.} \qquad (3.61)$$

Equation 3.61 is an equivalent discrete representation of Eqn. 3.57, except for the inclusion of the boundary and source terms. Therefore, the scheme is conservative. A similar proof may be developed for the viscous terms, showing that they also satisfy the conservation requirement.

3.7 Solver Implementation

Since unstructured grids do not use a logical array-like structure to provide the spatial relationships between mesh points, a separate accounting method is required to define this connectivity pattern. Unstructured grid methods normally maintain a list of the nodes which comprise each individual cell, as well as a connectivity array which lists the cells surrounding each node. In the current implementation, this usual construction is altered. Instead of applying the unstructured data model at the level of an individual cell, micro-blocks of $N \times N$ cells or $(N+1) \times (N+1)$ nodes

are arranged in small, locally structured arrays. These "patches" (or "chunks" in three dimensions [22]), are arranged in a globally unstructured fashion. By using this semi-structured approach, the connectivity array storage requirement is reduced and a number of additional advantages are found during grid adaptation (discussed in Chapter 5). This method of data decomposition also provides a natural format for the eventual implementation of a multigrid convergence acceleration technique. Successive levels of grid coarseness may be generated by recursively eliminating alternate grid lines in a patch. Extension of the method to parallel processing should be quite natural because the solution domain is already partially partitioned through the use of the micro-blocks. Implementation of both of these techniques is recommended for future extensions of this effort.

The two primary data structures used in the code are the CMESH array, which lists the nodes in a particular patch, and the CNEIB array, which lists the neighbor patches surrounding a patch. Globally maintained data, such as the dependent variables and various grid quantities, are stored in one-dimensional arrays. Entries in the CMESH array are pointers into these variable arrays; the position of a node in the CMESH array provides the logical structure of the patch. Figure 3.11 illustrates the entries in the CMESH array. An entry in a one-dimensional variable array is unique, implying that nodes on a patch boundary appear in two CMESH arrays along edges and in four CMESH arrays in the case of corner nodes. Using this data structure, the calculation of fluxes proceeds in a series of three phases: data gather, computation, and scatter. Data used in the flux calculation for a particular chunk are first copied (gathered) into local patch arrays, the required computations performed using these logically structured arrays, and the results copied (scattered) back into the global arrays. This methodology insures that indirect array references, which are a serious performance penalty, happen only twice, in the gather and scatter steps, and do not occur repeatedly throughout the computation phase.

global array
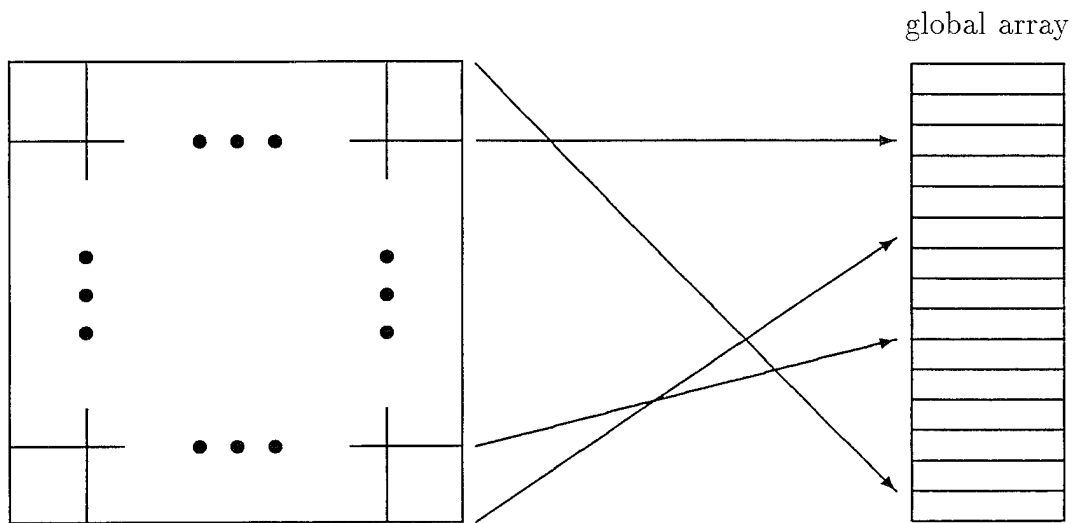
Figure 3.11: CMESH array pointer entries.

The other data structure, the CNEIB array, contains the connectivity pattern between individual patches; it is here that the "unstructured-ness" of the grid is maintained. An entry in the CNEIB array for a particular patch contains a list of the patches which share one of the four edges of the current patch. An edge which lies along a domain boundary receives an entry of zero in the appropriate location; edges along periodic boundaries will contain the number of the patch across the boundary. The information contained in this list is only required if adaptation of the grid is to be performed. Knowledge of the neighboring patches is used to account for a difference in grid adaptation level along a patch edge and to apply the various logical rules required during the adaptation process.

The solution domain boundaries are the only portions of the grid where lists of individual nodes are kept. Each boundary entry contains two elements: the first is the nodal index of the point on the boundary itself, the second is the index of the next point in from the boundary. These inner points are used for the extrapolations needed by many of the boundary conditions. Periodic boundaries differ from this format in that the two elements in the boundary entry are the numbers of the two nodes which share the boundary. Each patch also includes information about its

61

current adaptation level, the boundary condition type of each of its four edges, and an indication of an adaptation level change across the edge. Pointers are also kept to indicate the parent and children of a particular patch for use in the adaptation routines.

Chapter 4

TURBULENCE MODELING

Two different turbulence models have been implemented in the current work. The models are each briefly described below; further details may be found in the appropriate references. This description is followed by a discussion of the implementation-specific details of each model.

4.1 Algebraic Model

The algebraic turbulence model employed is a modified form of the Baldwin-Lomax algebraic eddy viscosity model [7] (referred to hereafter as B-L). Although this model was originally developed for steady flows, it has become a de facto standard for many applications because of the ease of its implementation and the minimal burden placed on computational resources. The eddy viscosity in the boundary layer is modeled using a two-layer formulation which requires a search normal to the wall in order to determine the location of the boundary between the layers. As is described below, this search causes a number of complications in an unstructured grid implementation.

In the inner layer of the boundary layer model, the turbulent eddy viscosity is given by the Prandtl-van Driest expression:

$$\mu_{T_i} = \rho(\kappa_1 y D)^2 \, |\hat{\Omega}|, \tag{4.1}$$

where

$$D = 1 - exp\left[-y\left(\frac{\rho_w \mid \hat{\Omega}_w \mid}{\mu_w}\right)^{\frac{1}{2}}/A^+\right].$$ (4.2)

$\mid\hat{\Omega}\mid$ is the vorticity magnitude which, for the quasi-3D formulation is given by

$$\mid\hat{\Omega}\mid = \frac{1}{2}\mid\frac{\partial v_\theta}{\partial m} - \frac{1}{r}\frac{\partial v_m}{\partial \theta} + v_\theta\frac{r_m}{r}\mid.$$ (4.3)

The variable y represents the distance normal to the nearest blade surface, A^+ is the sublayer thickness, and κ_1 is the von Karman constant. Jobe and Hankey [33] have shown that increasing A^+ and κ_1 improves the ability of the B-L model to predict turbulent boundary layers in adverse pressure gradients. Optimization of these parameters is beyond the scope of the current effort, and the model is implemented in its standard form with $A^+ = 26$ and $\kappa_1 = 0.4$.

In the outer region of the boundary layer, the turbulent eddy viscosity is given by:

$$\mu_{T_o} = \rho\kappa_2\,C_{cp}\,F_{max}\,y_{max}\,F_{KLEB},$$ (4.4)

where $F_{max} = max(y \mid \hat{\Omega} \mid D)$, $F_{KLEB} = [1 + 5.5(C_{KLEB}\,y/y_{max})^6]^{-1}$, y_{max} is the value of y where F_{max} occurs, $\kappa_2 = 0.0168$, $C_{cp} = 1.6$, and $C_{KLEB} = 0.3$. The turbulence model switches from the inner to the outer model at the first point at which $\mu_{T_i} \geq \mu_{T_o}$.

4.2   One-Equation Model

Past experience has shown that the B-L algebraic model overpredicts the separation region in the high-turning, high-contraction blade passages of modern transonic compressor designs [60]. An additional difficulty with the algebraic model is the tracking, through any subsequent rows, of the turbulent eddy viscosity generated

in upstream blade rows. The impingement of the upstream blade row wakes has a considerable impact on the flow field and performance of the downstream row, particularly in the high-speed cascade designs of interest. If an algebraic turbulence model is used, as in [36, 52, 60], it is unclear how the effects of an upstream blade row's eddy viscosity would be propagated into the downstream row. One- and two-equation models are well suited to this situation because convective and dissipative terms are included in the field equations, allowing the entire multi-blade row eddy viscosity field to evolve over time. Additionally, these models have a natural implementation in an unstructured grid environment; there is no need to perform a search out from the wall.

A number of one- and two-equation models are enjoying increased use as problems become more complex and the availability of faster workstations helps to ease computational limitations. The one-equation models of Baldwin and Barth [8], Johnson and King [34], and Spalart and Allmaras [71], and the two-equation models of Jones and Launder [35], Wilcox [79], Coakley [16], and Menter [48] are all being used throughout the external and internal aerodynamics communities. Each of these models have their relative advantages and disadvantages. The primary concerns in the current effort, beyond a model's predictive abilities, are its computational efficiency and robustness. During this work, the models of Jones and Launder, Coakley, Menter, and Spalart and Allmaras have each been implemented. However, the Spalart-Allmaras model has the advantage of a reduced equation set and has given the least difficulty in its use.

The Spalart-Allmaras (S-A) model has a number of advantages that make it particularly suitable for an unstructured grid implementation. The first is that the model is "local" in the sense that the solution at one point does not depend on that at other points. This is contrasted with the B-L model which requires a search away from the wall in order to locate a local maximum. The model does not depend on

the use of length scales, which increases its generality and applicability to a wider variety of flows without case-specific parameter tuning. Finally, it has the distinct advantage of not requiring grids which are significantly finer than those required to capture a turbulent velocity profile. This implies an ability to use coarser grids than those required by other one- and two-equation turbulence models.

The S-A model uses a single transport equation for the turbulent eddy viscosity. The turbulent viscosity is given by:

$$\nu_T = \tilde{\nu} \left[\frac{\chi^3}{\chi^3 + c_{v1}^3} \right],$$

(4.5)

where $\chi \equiv \tilde{\nu}/\nu$, and $\tilde{\nu}$ is the dependent variable of the following transport equation:

$$
\begin{aligned}
\frac{D\tilde{\nu}}{Dt} &= c_{b1}\tilde{S}\tilde{\nu} - c_{w1}f_w \left[\frac{\tilde{\nu}}{y} \right]^2 \\
&\quad + \frac{1}{\sigma} \left[\nabla \cdot \left((\nu + \tilde{\nu})\nabla\tilde{\nu} \right) + c_{b2}(\nabla\tilde{\nu})^2 \right].
\end{aligned}
$$

(4.6)

In this equation,

$$
\begin{aligned}
c_{w1} &= c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma, & (4.7) \\
f_w &= g \left(\frac{1 + c_{w3}}{g^6 + c_{w3}} \right)^{1/6}, & (4.8) \\
g &= r + c_{w2}(r^6 - r), & (4.9) \\
r &\equiv \frac{\tilde{\nu}}{\tilde{S}\kappa^2 y^2}, & (4.10)
\end{aligned}
$$

and

$$
\begin{aligned}
\tilde{S} &\equiv |\hat{\Omega}| + \frac{\tilde{\nu}}{\kappa^2 y^2} f_{v2}, & (4.11) \\
f_{v2} &\equiv \left(1 + \frac{\chi}{c_{v2}} \right)^{-3}. & (4.12)
\end{aligned}
$$

66

As before, $|\hat{\Omega}|$ is the vorticity magnitude and y is the distance to the nearest wall. These equations are those proposed in the original S-A model. In their modeling of dynamic stall in airfoils, Ko and McCroskey [38] made a number of modifications to the model in order to improve convergence by preventing $\tilde{S}$ from becoming locally negative. After repeated difficulties with instabilities, the Ko and McCroskey version was abandoned in favor of the original implementation. As suggested in [71], limiters have been included on the value of r to insure that it remains in the range $0 - 10$.

Constants for the original model given in [71] are $c_{b1} = 0.1355$, $c_{b2} = 0.622$, $\sigma = 2/3$, $c_{w2} = 0.3$, $c_{w3} = 2$, $c_{v1} = 7.1$, $\kappa = 0.41$ and $c_{w1} = c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma$. While the original S-A model provides additional terms for the prediction of the transition location, these terms are omitted here; the flow fields are considered to be fully turbulent.

The boundary conditions which were implemented in all cases include a homogeneous condition at the wall surface, the specification of the freestream value at the inlet plane, and an extrapolation of the viscosity parameter at the exit. A nominal freestream value of $\tilde{\nu} = \frac{\nu}{10}$ is used, however, neither the solution converged values, nor the stability of the solution during transients, were sensitive to this value.

4.3   Implementation of the Models

The following section discusses the implementation-specific factors involved with each model. Validation examples are presented in Chapter 6.

4.3.1   Algebraic Model

As previously described, the B-L algebraic model requires a search normal to the wall in order to determine the boundary between the inner and outer models. This

search causes difficulties in a pure unstructured implementation because of the computational overhead in determining nearest node information. Some authors have simply ignored the problem by using a structured grid adjacent to the blade surface and including the turbulence model and the turbulent eddy viscosity only in the inner grid [62, 69]. Other authors have overcome this difficulty by either overlaying a structured grid near the blade surface [46], or by maintaining a search tree data structure to facilitate the search for the inner-to-outer model transition location [37]. The latter method is used in the current effort. A search line, called a "thread," is generated and stored for each surface point after each grid adaptation cycle. Since the turbulent viscosity is updated only occasionally (every 50 iterations for steady flow cases), these threads are calculated and written to a temporary disk file. Storing the threads allows the search for the inner-to-outer model transition to easily be performed along lines that can be traced directly back to the blade surface. The turbulent eddy viscosity at all those points which are not directly connected to a blade surface (due to grid adaptation), is calculated by a recursive interpolation from points along the threads. A drawback of the current implementation is the lack of a specific wake model, which would track the wake centerline downstream of the trailing edge.

From past experience, the B-L model tends to overpredict regions of separated flow. Experience has shown that much of this is due to "noise" in the search away from the wall for the value of F_{max}. A modification was introduced to only include values in this search which have a positive streamwise velocity component. Ignoring regions of reverse flow causes the model to yield larger values of eddy viscosity in these areas, helping to suppress the formation of flow separation.

4.3.2 One-Equation Model

The one-equation model uses an equation set which is quite similar to that solved for the flow equations, and is therefore solved using an identical algorithm. An n-stage Runge-Kutta is used to update the turbulent equations before each iteration of the flow equations. The code has been implemented so that different Runge-Kutta schemes may be used for the flow and turbulence equations. This general method also allows the use of sub-iterations for the turbulence equations, which may require smaller time steps than the flow equations. In a treatment similar to that used for the flow equations, the convective and source terms are evaluated during each stage of the Runge-Kutta step, while the diffusive terms are updated during the first stage and then frozen for the remaining stages. Second- and fourth-order damping is included in the Runge-Kutta scheme, with the coefficients ϵ_2 and ϵ_4 of Eqn. 3.23 taken directly from the previous iteration of the flow equations. The damping required is typically quite small. Extensive numerical testing has shown that damping with coefficients on the order of $10 - 20\%$ of those used for the flow equations is adequate to suppress formation of any sawtooth modes. For coarse grid solutions, a single iteration of the same 5-stage Runge-Kutta scheme used for the flow equations is used. For adapted solutions, sub-iterations of the turbulence model are occasionally required to prevent solution instabilities. These instabilities typically occur in highly loaded compressor calculations, forming near the trailing edge and causing rapid solution divergence. At most, two sub-iterations of the 4-stage scheme are needed. Additionally, a limiter is placed on the viscosity parameter to insure that it does not become negative; detected values less than zero are reset to one half of the freestream value.

One concern in the implementation of the turbulence model is the calculation of the wall distance function for rotor-stator interaction cases. For an isolated cascade
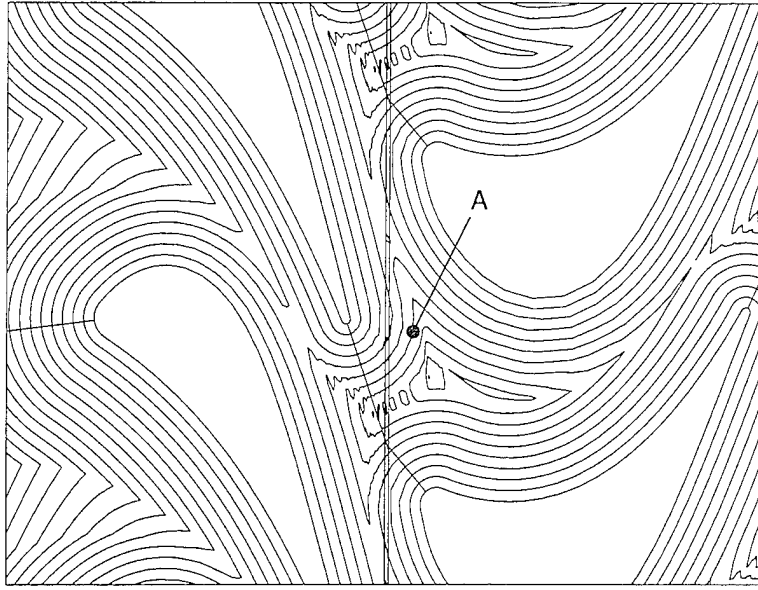
Figure 4.1: Contours of wall distance function in a closely coupled turbine stage.

computation, the wall distance does not change and can simply be computed once and saved. For a closely coupled stage, there may be situations where the closest surface point to a particular node is on a blade in a different row than that in which the node is located. An example is shown in Fig. 4.1, where point A lies in the downstream blade row, but it is actually closer to the surface of the upstream blade at this particular moment in time. Although it is possible to update the wall distance function as the grids move past each other, the turbulence model is sensitive to wall distance only for those points that fall within the logarithmic region of the boundary layer [9]. Therefore, as in isolated airfoil calculations, the wall distance is computed at the beginning of the computation and stored.

Chapter 5

ADAPTATION TECHNIQUES

One of the major benefits of unstructured grid methods is the ability to locally refine the computational grid in order to introduce additional grid points in regions where increased resolution is desired. This chapter describes the generation of the initial, or base grid, the procedure for adapting the grid, and the method in which flow features are detected. Finally, the algorithm used to account for the midface nodes created at the adaptation interface is described.

5.1   Cascade Meshes

All adaptive grid flow solvers require the generation of an initial coarse mesh from which the convergence and adaptation process proceeds. Unstructured grid methods that use triangular cells often generate the initial grid using the adaptation algorithm, with the distance to solid surfaces as a weighting function. Quadrilateral cell methods generally use structured grid generators to provide the initial coarse mesh, usually in the form of H-grids for cascade calculations.

A difficulty with using H-grids, however, is the poor resolution of leading and trailing edges with significant curvature. The flow field through a transonic compressor cascade, and particularly the accurate determination of shock losses, is strongly dependent on proper calculation of the strength of the leading edge bow shock. In order to provide better resolution of the leading and trailing edge radii in the transonic cascades used in the current effort, an O-grid is wrapped around the

blade and blended into an H-grid in the passage. This blending can be performed in two ways. One method is to use a single H-grid in the blade passage, which produces acceptable grids, but can cause excessive grid skew up- and downstream of the blades. To improve the grid quality, additional H-grid blocks can be introduced up- and downstream of the O-grid. Figure 5.1 shows a sample of both of these grid types at the leading edge of typical transonic compressor cascade. The unstructured grid is generated using a very flexible, interactive graphical grid generator which allows a great deal of user control over all grid parameters. A description of the grid generator is provided in Appendix C. This gridding tool may be used to generate either the simple or the multi-block form of the blended O-H grids.

Each of these blended grid types create points which are surrounded by either five or six cells rather than the usual four. These special points have a minor effect on the flow solver algorithm, requiring special treatment during both the flux integration and the adaptation process. This overhead is more than compensated for by the reduction in grid skewness near the blade surface, and the ease in which the solver may be adapted to cascades with blunt leading edges. Points surrounded by more than four cells require a modification of the inviscid flux terms to account for the presence of the extra cells. During the flux summation, these special nodes receive the usual $\frac{1}{4}$ distribution of fluxes from each of the cells around the point. This is followed by a multiplication of the nodal flux value by either $\frac{4}{5}$ or $\frac{2}{3}$ depending upon whether there are five or six cells around the node, respectively.

5.2   Mesh Refinement

The concept behind spatial grid adaptation is to adjust the local grid density to match the resolution required by the flow. This involves increasing the grid density in areas where high resolution is required, such as near shocks and in boundary
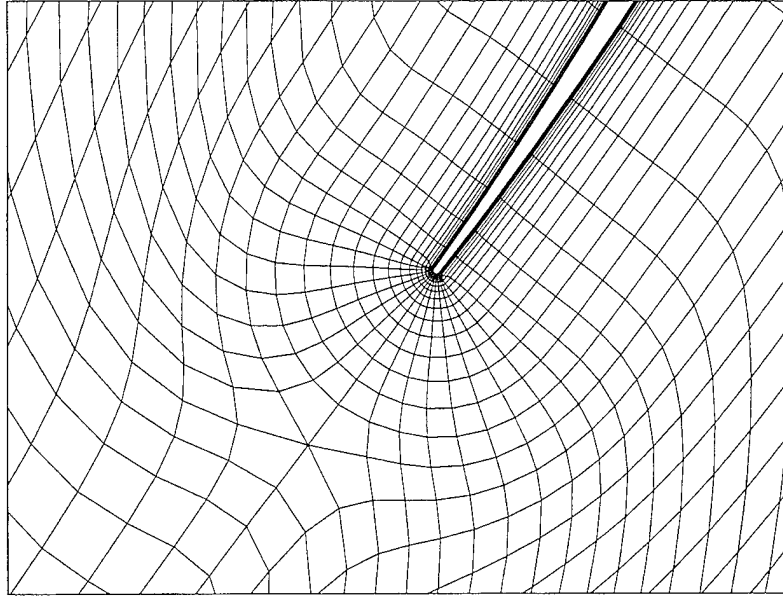
(a) Simple O-H grid
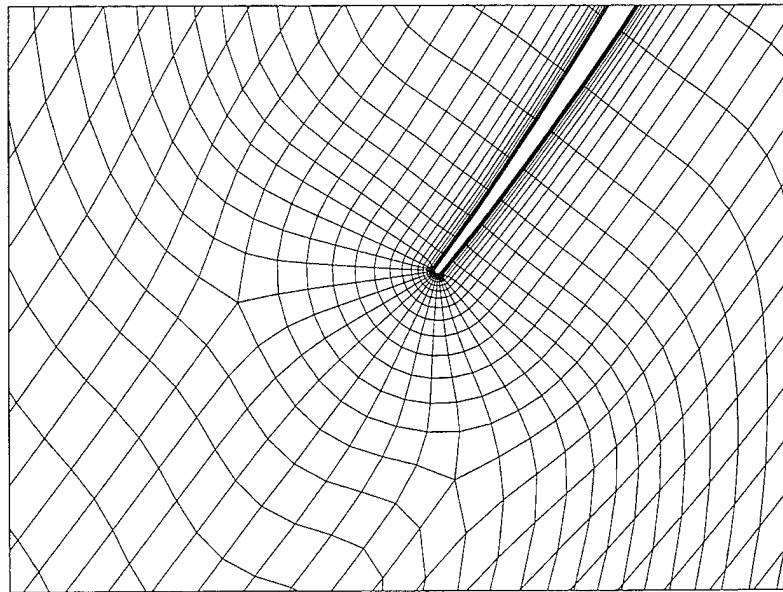
(b) Multi-Block O-H grid

Figure 5.1: Leading edge details of blended O-H grids.

layers, and remove grid points from areas of nearly uniform flow. Grid adaptation has historically followed two paths. The first is to retain the original mesh connectivity and only redistribute the existing grid points, "attracting" them to regions of high flow gradient [11, 20]. This method has the obvious drawback that the grid remains bounded by the initial number of grid points. Additionally, it is often difficult to sufficiently manipulate the grid to track multiple, complex flow features. An alternate approach is to locally embed additional cells in those regions of the flow requiring increased resolution. This allows the grid to completely match the complexity of the flow features with a minimum number of points.

Adaptation by refinement may be further divided into two classes: grid regeneration and cell division-recombination. Grid regeneration, which is the technique often used for triangular element-based unstructured methods, uses a method such as the advancing-front technique of Peraire et al. [50], to form a new grid using flow field features to determine the local grid resolution. While this requires no modification of the solver algorithm for the adapted grid, the disadvantage of this method is the time required to regenerate an entire grid. For unsteady solutions, the grid is adapted frequently and the adaptation method must not produce significant overhead.

With the cell division method, which is used in the current effort, single cells are simply divided in one or both directions depending on the value of the flow gradients. In triangular element meshes, cell division must often be followed by a local re-triangulation of surrounding cells in order to minimize the amount of grid skew [9].

Cell division in quadrilateral elements results in the creation of midface nodes at the boundary between successive levels of grid refinement. These midface, or hanging, nodes require special treatment in the evaluation of the flux integrals (described in Section 5.4 below). The adaptation procedure also needs to avoid
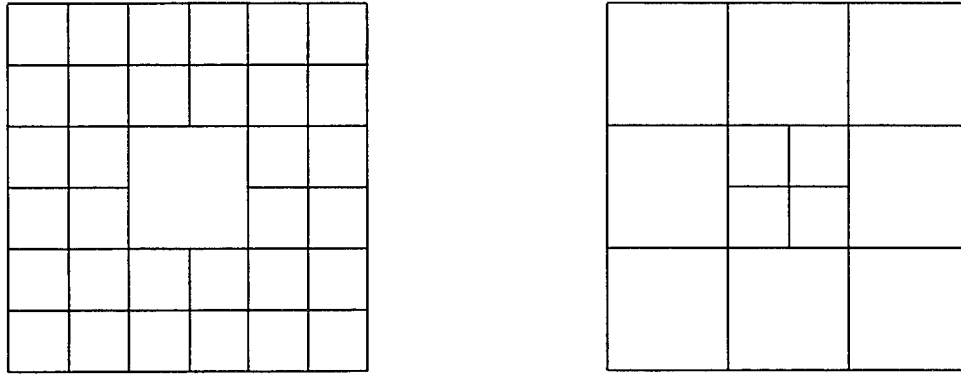
74

Figure 5.2: Void and island cells.

the generation of cell faces with more than a single midface node. This is prevented by simply ordering the cells to be divided from coarsest to finest adaptation levels.

An advantage of the semi-structured grid approach used in the current effort, over a standard purely unstructured grid method, is the sizable reduction in the additional logic required during adaptation. In a purely unstructured implementation, such as [2, 61], rules are applied to eliminate the generation of any undivided cells completely surrounded by divided cells ("voids") or isolated divided cells ("islands"), as illustrated in Fig. 5.2. While in some cases, removal of these void and island cells is required by the data structure [61], they are generally undesirable because of the overhead required by the midface nodes. In the semi-structured approach with patches containing more than one cell, neither voids or islands may exist in the mesh, greatly reducing this adaptation overhead.

A standard method of locating new grid points in an adapted cell is to place them at the midpoints of the previous cell faces. Edge nodes use the data from the neighboring corner nodes, while nodes added at a cell's center use the average of the four corner nodes. Another advantage of the semi-structured approach lies in the ability to maintain the grid stretching distribution during the division of a patch. Within a patch, it is possible to determine the stretching ratio of the nearby cells
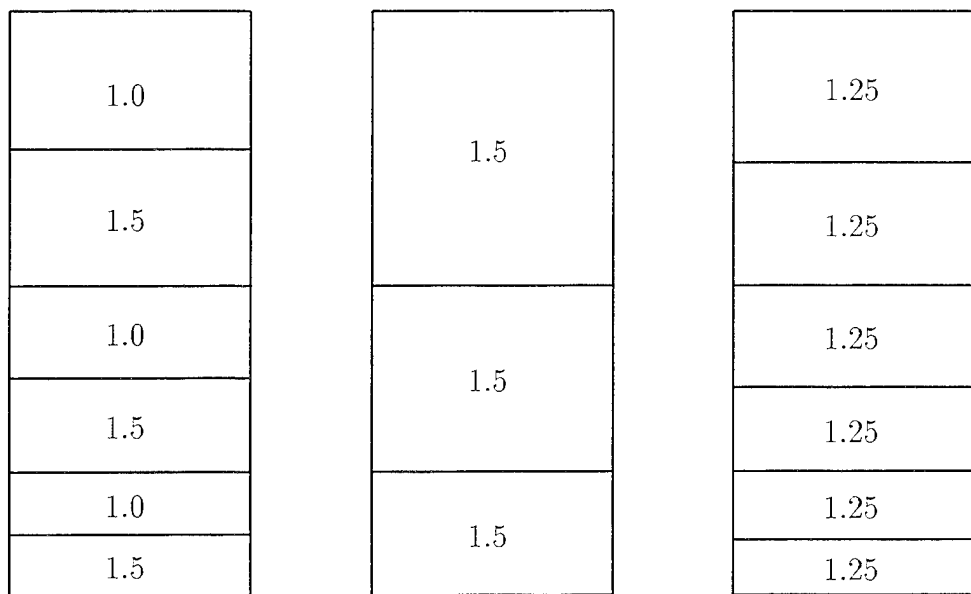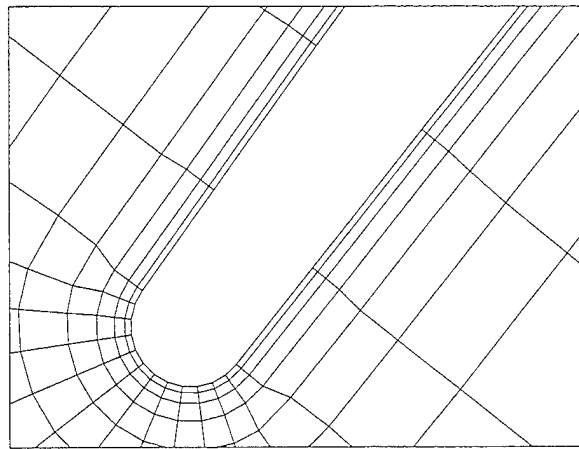
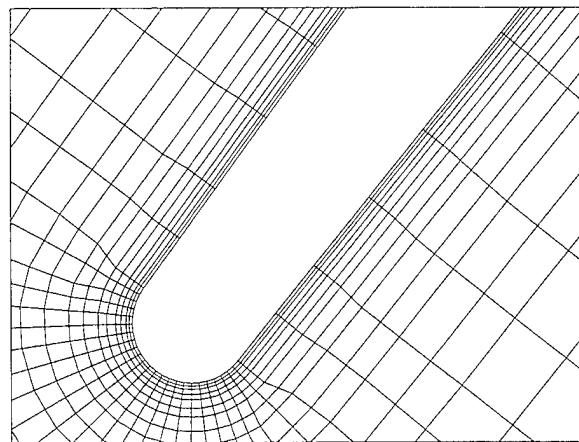(a) simple cell division   (b) original grid   (c) smooth cell division

Figure 5.3: Variation in stretching ratio for two forms of cell division.

and place new node points within the patch in accordance with this pre-existing stretching. This methodology allows the production of refined cells with stretching ratios which are half those of the parent cells, thereby reducing stretching error as the grid is refined. The improvement in stretching ratios between the current method and the use of simple cell division is depicted in Fig. 5.3. The numbers in each cell represent the stretching ratio between that cell and the one immediately below it. Grids used in actual computations are often quite stretched around the leading edge and the use of this alternate method of locating new points produces a rather dramatic improvement in the grid smoothness, as seen in Fig. 5.4

Points on the blade surfaces are determined by an interpolating from a geometric data set substantially finer than the initial grid. When interpolating surface points in viscous grids, which have high cell aspect ratios near solid walls, regions of high surface curvature are likely to cause incorrect placement of points within the

Unadapted Grid

Simple Cell Division
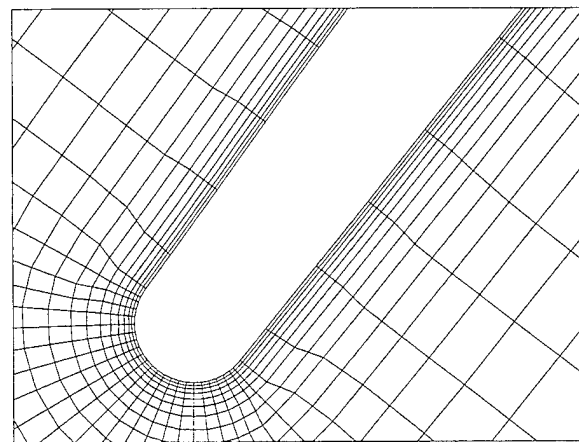
Smooth Cell Division

Figure 5.4: Improvement in actual grid by using smooth cell division.

blade surface, as illustrated in Fig. 5.5. For this reason, points along added normal grid lines are relocated, smoothly moving them out from the wall to remove this error condition. Points to be shifted are identified in a manner similar to that used to create the "threads" used in the algebraic turbulence model; generally no more than ten points must be shifted to avoid the generation of these incorrect cells.

5.3 Feature Detection

One of the major difficulties to be overcome in the use of any adaptive grid solver is the identification of the flow field feature to which the grid is to be adapted. Generally some a priori knowledge of the flow field is necessary in order to determine which features are of importance and what function should be used to locate them. Static pressure gradient is typically useful for locating shocks, but will not detect boundary or shear layers. Mach number and velocity gradients indicate both shock waves and shear layers, but the large velocity gradient in shear layers often masks the detection of shock waves, particularly in initial adaptation cycles when shocks are highly smeared. Density gradient is often a good compromise because both shocks and shear layers are detected. Another technique, which has been found to be quite useful in the current effort, is flow feature detection based on the gradient in the computed turbulent eddy viscosity. This approach allows resolution of the wake region further downstream of the trailing edge than with the use of velocity gradient alone. This section discusses the detection of both strong and weak features, and considers the special case of adaptation to unsteady flows.
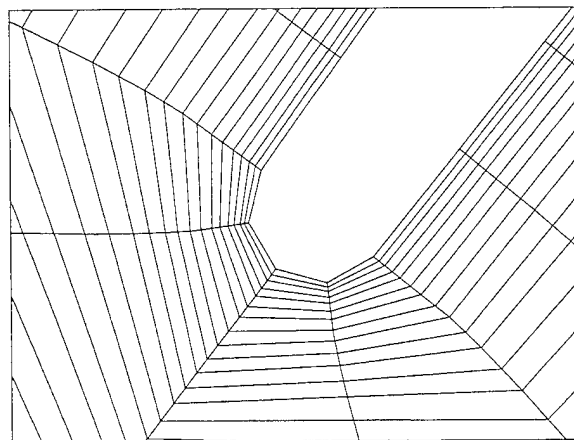
5.3.1 Strong Features

One of the primary reasons to adapt the computational grid is to improve the local grid resolution near primary flow features. Of the possible flow phenomena
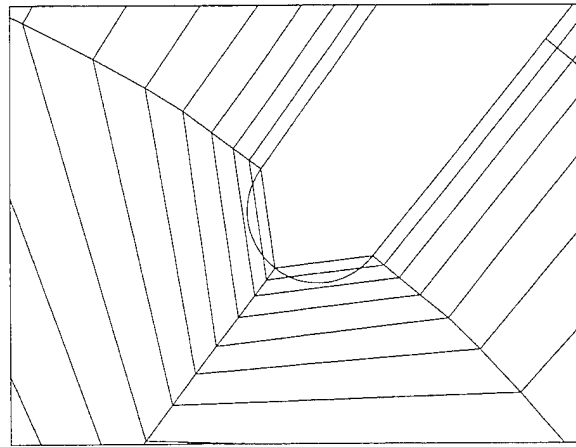
**Unadapted Grid
Overlaid with
Blade Data**
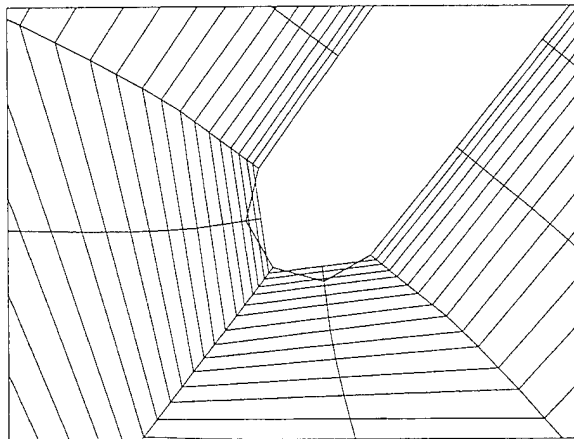
**Without
Shifting**

**With
Shifting**

Figure 5.5: Requirement for grid shifting after surface fit.

in the transonic flows considered in this thesis, the only true discontinuities in the flow are shock waves. Physical shocks have a thickness on the order of several mean free paths, far beyond the resolution of any typical CFD code, and would require continued grid refinement far beyond a level which might be considered "adequate." Additionally, if grid adaptation were based solely on the statistical approach described below, shocks would tend to skew the statistics and mask many of the smoother features. For this reason, a separate pass is made through the solution domain to detect those patches which contain shocks. Shocks are found by marking those patches which have a normalized second difference in pressure above some user-specified threshold. This pressure difference is the same as the pressure switch used in the fourth-order damping, give by Eqn. 3.24. Threshold values are generally set at $\Delta P \geq 0.02$, although the detection algorithm is somewhat insensitive to the exact value. An example of a transonic compressor cascade pressure field and the detected shock patches is shown in Fig. 5.6.

Another feature detection method which has proven useful, detects those patches along the wall that have a y^+ value at the first point away from the wall below some threshold. For turbulent boundary layers, which have high surface velocity gradients, there is a certain point beyond which continued resolution does little to improve the solution. Adding more cells near the wall serves only to produce cells which further limit the maximum time step that may be taken by the solution algorithm. Currently, this threshold value is set at a $y^+ = 1$. In practice, this limiter does remove some wall patches, but adaptation often occurs in the next layer of patches away from the wall, which usually contains the knee of the velocity profile.

5.3.2 Smooth Features

Once the patches containing strong features are removed from consideration, a sweep is made through the remaining patches to adapt the grid in regions of smooth, but

Static Pressure
Distribution
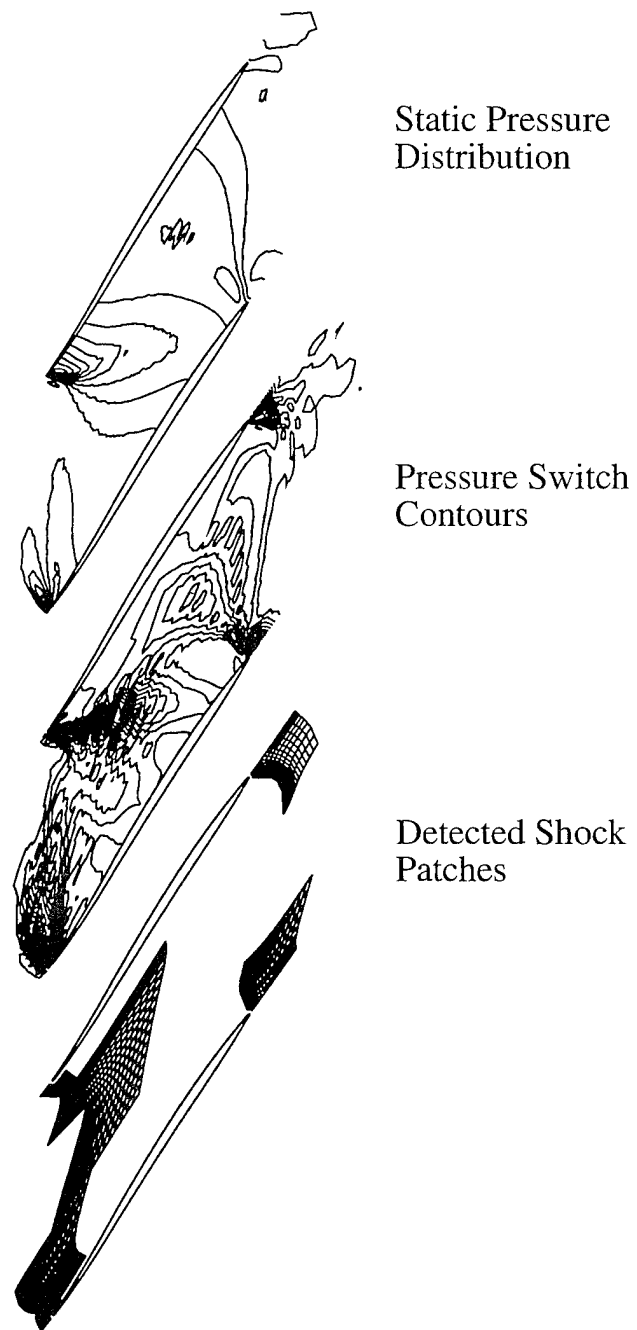
Pressure Switch
Contours

Detected Shock
Patches

Figure 5.6: Transonic fan cascade pressure field and detected shock patches.

rapid, flow variation. The method used triggers cell division and recombination using the undivided differences of tow user-specified parameters, usually density and Mach number or eddy viscosity. Threshold values for cell division and recombination are determined based on the statistics of the gradient field, as described by Aftomis and Kroll [2].

In the semi-structured implementation used in the current effort, an entire patch is adapted rather than a single cell. Therefore, the decision to adapt must be made by considering the aggregate flow gradient throughout the patch. In practice, once the patches containing strong features are removed, a sweep is made through the remaining patches to determine the local mean and standard deviation of the trigger function in each patch. These are combined to form a global mean and standard deviation and, from these, the upper and lower threshold values. For a patch to adapt, it is not necessary that every cell within the patch be tagged for division or recombination. Instead, some percentage of the cells within the patch must lie above or below the trigger value. When adapting to two quantities, division occurs when either gradient exceeds the divide threshold; recombination only occurs when the gradient for both quantities is less than the threshold value. Local patch threshold values are determined by using the local values of the mean and standard deviation. For example, a patch will divide if its local threshold is above some globally determined value:

$$T_{\substack{local \\ divide}} \geq T_{\substack{global \\ divide}}, \tag{5.1}$$

where

$$
\begin{aligned}
T_{\substack{local \\ divide}} &= \overline{\Delta f}_{patch} + C_{local}\sigma_{patch} \\
T_{\substack{global \\ divide}} &= \overline{\Delta f}_{global} + C_{divide}\sigma_{global}.
\end{aligned} \tag{5.2}
$$

82

Here, $\overline{\Delta f}$ indicates the average value of the trigger function f, and σ indicates its standard deviation. C_{local} is a factor determined from the mathematics of a normal distribution curve and sets the statistical percentage of cells within a patch that will divide. C_{divide} is the user specified threshold value based on the global quantities. From simple statistics theory, the expressions for the mean and standard deviations in both the patch and the global domain are given by:

$$\overline{\Delta f}_{patch} = \frac{\sum_{\substack{patch \\ cells}} (\Delta f)}{N_{\substack{patch \\ cells}}}, \tag{5.3}$$

$$\sigma_{patch} = \sqrt{\frac{\sum_{\substack{patch \\ cells}} (\Delta f^2) - N_{\substack{patch \\ cells}} \left(\overline{\Delta f}_{patch}\right)^2}{N_{\substack{patch \\ cells}} - 1}}, \tag{5.4}$$

and

$$\overline{\Delta f}_{global} = \frac{\sum_{\substack{all \\ patches}} \left(\overline{\Delta f}_{patch}\right)}{N_{patches}}, \tag{5.5}$$

$$\sigma_{global} = \sqrt{\frac{\sum_{\substack{all \\ patches}} \left(\sum_{\substack{patch \\ cells}} (\Delta f^2)\right) - N_{patches} \left(\overline{\Delta f}_{global}\right)^2}{N_{patches} N_{\substack{patch \\ cells}} - 1}}. \tag{5.6}$$

Generally, division is allowed when 30% of the cells within a patch exceed the division threshold, yielding a $C_{local} \approx 0.5$. Similar expressions can be written for cell recombination, requiring the specification of a C_{fuse} parameter. Normally, 80% of the cells in a patch are required to fall below the fusion threshold before the patch is allowed recombine. Because of the obvious skew of the distribution in a typical adaptation function histogram, as seen in Fig. 5.7, the C_{divide} is usually set at about 0.8, while C_{fuse} is normally set at 0.3. Note that the histogram in Fig. 5.7 is shown after the smooth feature detection, but before the application of the adaptation rules. Many of the patches that satisfy the feature detector threshold requirements will not fuse or divide because they violate these additional rules.
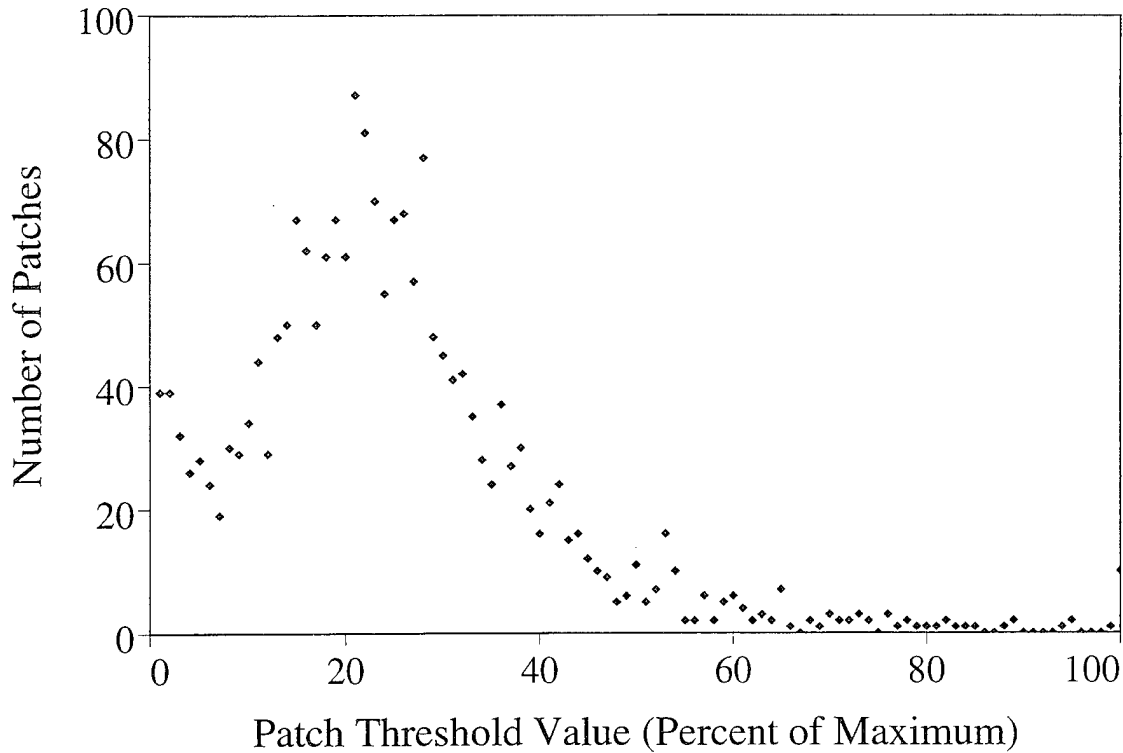
83

Figure 5.7: Typical adaptation function histogram.

5.3.3 Unsteady Flow Detection

Detection of flow features in unsteady flows poses an additional challenge because either the grid, the flow features, or both are moving. The object is to keep the grid refined both at the current location of a flow feature, as well as during the series of iterations before the next adaptation. Typically, the method used to insure adequate resolution is to lower the division threshold somewhat, and to adapt the grid frequently. This adaptation must be balanced by the increase in computational time due to the added number of cells to be processed. One method is to adjust the maximum number of adaptation levels to control generation of excessive cells.

An additional method is to set a upper limit on the number of cells that can be generated at a given adaptation level. The cells allowed to adapt are ranked by their adaptation function value and the threshold for grid division adjusted to restrain
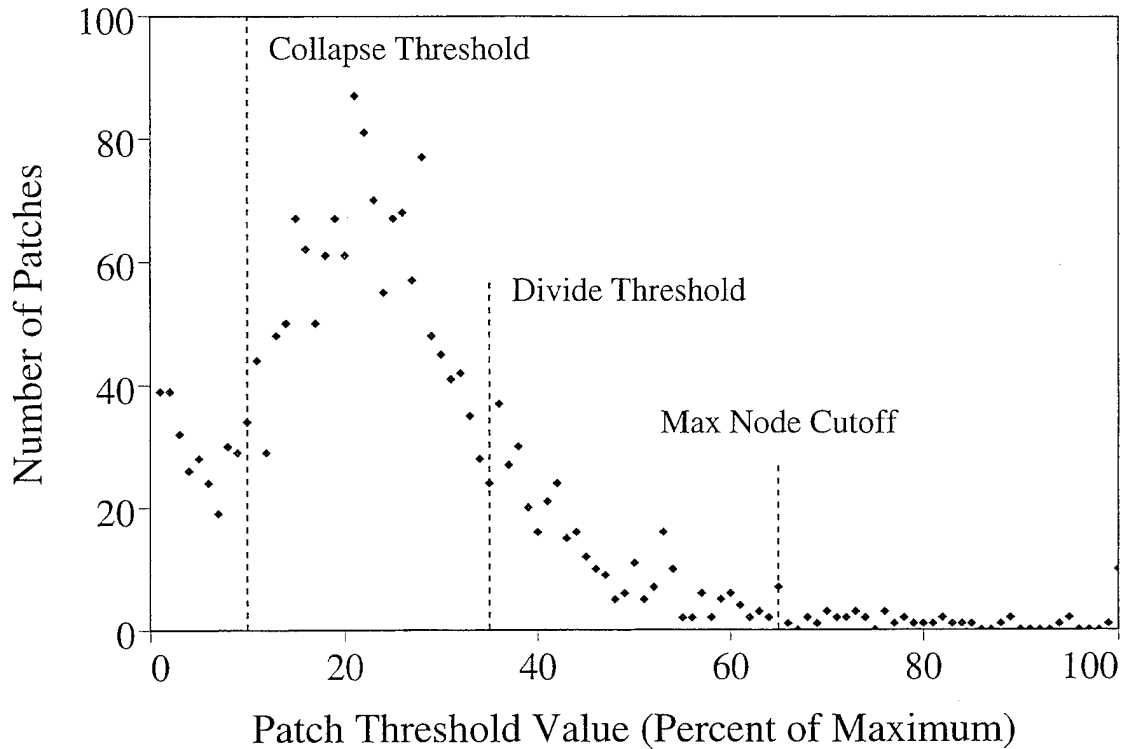
Figure 5.8: Adaptation function histogram with cell division and recombination thresholds.

the continued growth of the adapted solution. Figure 5.8 shows a typical adaptation function histogram with the divide and fuse thresholds, as well as the cut-off value used to prevent excessive cell production.

5.4   Numerical Treatment of Midface Nodes

The most difficult obstacle to the grid adaptation is the proper treatment of the interface region between patches of different adaptation levels. The integration scheme must maintain both conservation and accuracy at the interface, while not imposing an undo computational burden. This section will examine the problem in more detail and describe the algorithm that has been implemented.

The difficulty at the interface region is due to two factors: a sudden change in the grid stretching, or *metric discontinuity*, and the actual discontinuity in the grid

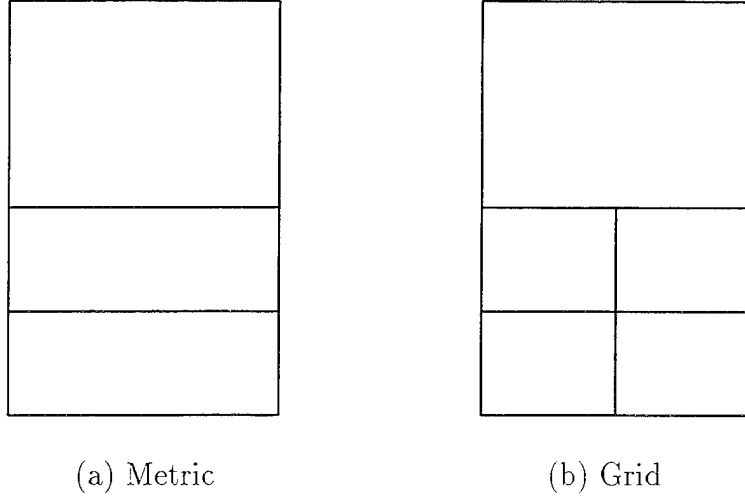(a) Metric   (b) Grid

Figure 5.9: Types of grid interface discontinuities.

itself caused by the midface nodes (see Fig. 5.9). The first of these causes a significant stretching error, which may harm the overall order of accuracy of the method. On the other hand, the inclusion of midface nodes requires special integration procedures to insure that conservation is maintained.

There are typically two methods employed to perform the flux integration at the boundary between cells of different adaptation levels. The first method includes the hanging nodes in the integration of the coarse grid cells with a custom integration stencil. Figure 5.10 shows the integration distribution stencils for a standard cell and for cases of cells with one or two hanging nodes. (The semi-structured approach prevents any cells from having more than two hanging nodes for patches containing more than one cell). This treatment has been shown to be conservative, but to suffer from an excessive stretching error [19, 37]. This interface method also presents a number of difficulties in its implementation in the code. Hanging nodes are only contained within the nodal definition array of the fine level patch and are not part of the coarse patch data structure. Therefore, inclusion of these hanging node terms in the coarse cell integration would require unwanted communication beyond the boundary of a single patch. A form of this method has been

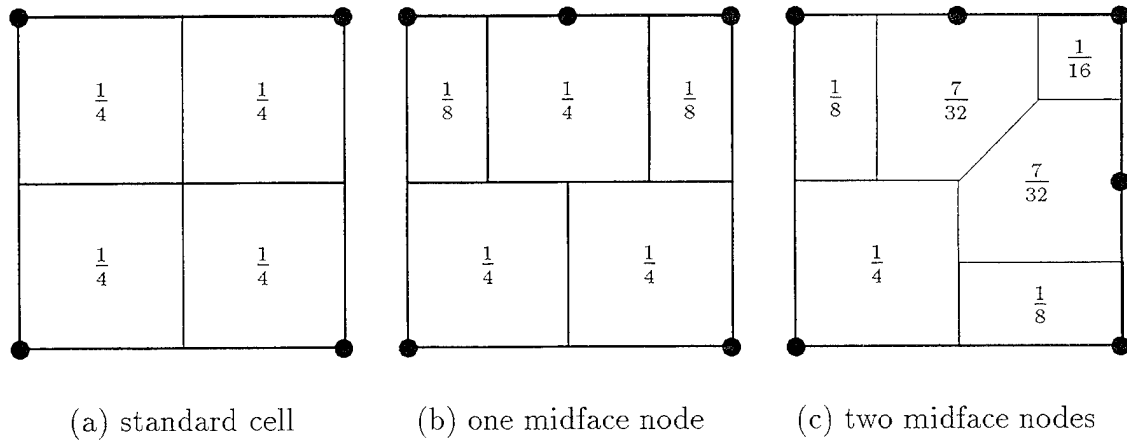(a) standard cell   (b) one midface node   (c) two midface nodes

Figure 5.10: Midface node cell integration stencils for conservative interface approach.

implemented by assuming that these hanging nodes have a value derived from an average of values of the neighboring nodes along the edge. This requires a secondary sweep through those coarse cells that border finer cells to correct the fluxes in the four corners of the coarse cell, without modifying the coarse cell flux contribution to the hanging node. This method allows the use of data available within a patch, without requiring communication to the surrounding patches. A final pass is needed to update all hanging nodes with the average of their neighbors. Use of this conservative interface approach is only of value in the evaluation of the inviscid fluxes, which are more susceptible to conservation errors. Calculation of viscous fluxes is quite sensitive to stretching error and therefore requires use of an interface method that maintains accuracy at the expense of conservation.

The operations in the conservative treatment may be summarized as follows:

- Complete a normal flux integration sweep through all cells in the patch.

- Along each edge that borders a finer cell:

  1. Calculate the midface node value by averaging the neighboring node values.

2. Calculate the change in the cell flux resulting from this midface node.

3. Using the modified distribution stencils, distribute this change to the corners of the coarse cell.

● After the global variable update, calculate the hanging node dependent variables by averaging the values at the adjacent nodes.

The accurate approach treats the nodes along the interface as if they belonged only to the coarse cell, eliminating the stretching error that exists in going from a coarse to fine cell. A "supercell," which is a combination of four fine grid cells as shown by the dashed box in Fig. 5.11, is used to perform this updated integration. The fluxes calculated during this secondary pass replace the fine cell fluxes at the coarser level cell nodes along the interface originally found by considering the fine cells only (i.e., only nodes A and C in Fig. 5.11). This treatment ignores the presence of the hanging nodes in the flux integration of coarse level cells along the patch boundary. However, these nodes contribute to the fluxes in the fine level patch cells. Once the nodal fluxes have been determined and the dependent variables are updated, the hanging node variable values are determined by averaging the variables at the adjacent nodes. There are three choices for the selection of this averaging of neighboring nodal values. The first, which is to perform a simple average of the dependent variable quantities, must be discarded because of the nonlinearity of the flow equations. The second choice, which would allow the accurate treatment to also be conservative, is to use the dependent variable values resulting from an average of the flux vector values (F or G in Eqn. 2.16). Unfortunately, this would require the solution of an equation which is of fourth order in the dependent variable values! The final choice is to average the primitive variables, (ρ, v_m, v_θ, p), which alleviates the nonlinearity problem and is computationally acceptable.
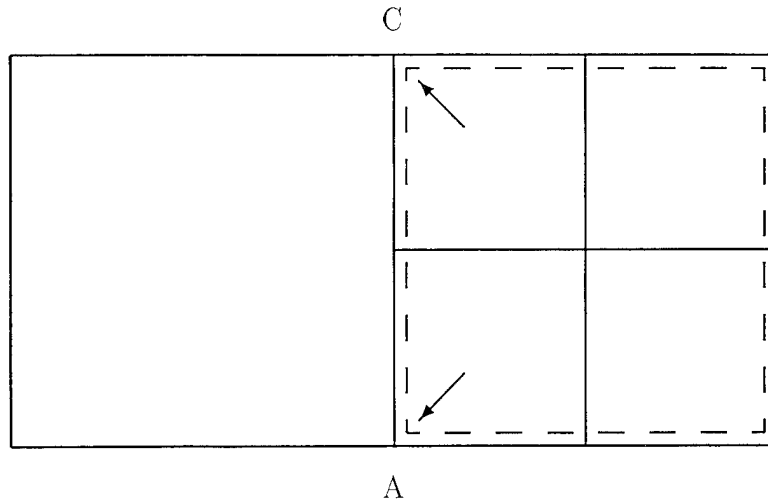
C

A

Figure 5.11: Formation of "supercell" for accurate interface treatment.

The operations in the accurate treatment may be summarized as follows:

- Complete a normal flux integration sweep through all cells in the patch.

- Along each edge that borders a coarser cell:

    1. Re-initialize the flux values of the coarse nodes.

    2. Perform a flux integration in the "supercells" along this edge.

    3. Distribute the calculated fluxes to the coarse nodes on the edge only.

- After the global variable update, calculate the dependent variables at hanging nodes by averaging the values at their adjacent nodes.

A similar operation is performed in the calculation of the second- and fourth-order smoothing terms and is also included in the updates of the one-equation turbulence model. In addition, it was also found necessary to perform the hanging node averaging of the turbulent eddy viscosity.

A final note in the implementation of the accurate treatment is the importance of detecting and treating inside corners between adaptation levels, as shown
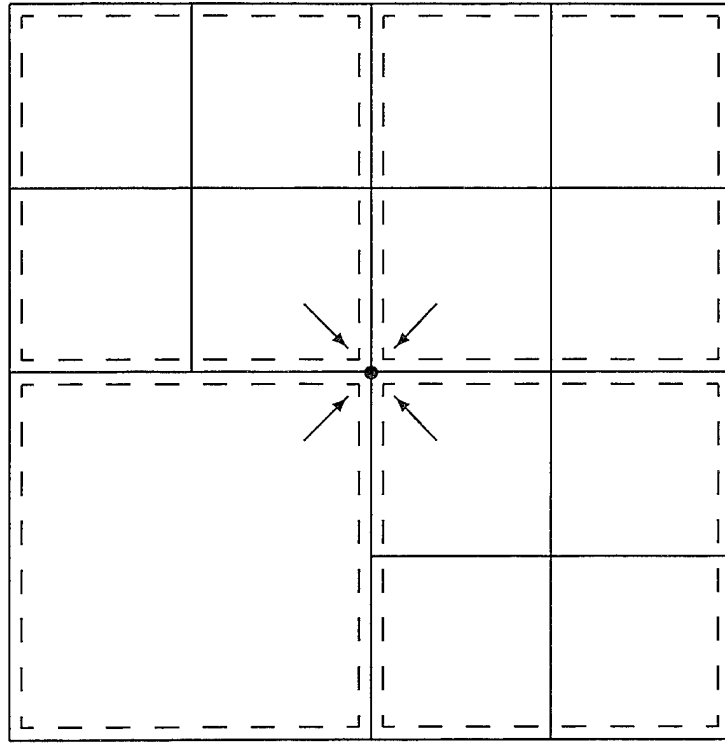
Figure 5.12: Accurate treatment of inside corner nodes taking into account all surrounding coarse level "supercells."

in Fig. 5.12. The aim of the accurate treatment is to update the coarse grid nodes along an interface boundary with an integration stencil fully at the coarse grid level. Therefore, logic has been introduced into the code to detect inside corners and apply the coarse grid integration stencil to the fine grid level patches that share these corner nodes. When two adjacent edges of a coarse grid level patch are found to be divided, a search is made around the corner node to tag all the patches sharing this node. This allows the algorithm to detect inside corner nodes which occur at the special grid block intersection points with five or six surrounding cells.

Chapter 6

RESULTS

This chapter presents results from the use of the computational model described in the previous chapters. Results are first presented for some of the simple test cases used to validate the solver. Computations of steady cascade flows in transonic turbomachinery are then given, followed by an example case using the unsteady adaptive solver.

6.1 Solver Validation

A variety of validation exercises were performed in order to evaluate various portions of the computational model separately. Unless otherwise stated, all basic test cases are performed with unit radius and streamtube thickness. In steady flows, convergence is typically assumed once the L^2 norm of all dependent variable residuals has fallen four orders-of-magnitude. When using the dual time stepping procedure for unsteady flows, each step in real time is allowed to converge two orders-of-magnitude. Unsteady periodic flows are converged when an integrated quantity, such as lift or drag coefficient, has become periodic. Non-periodic unsteady flows are assumed to be converged at each real time step as the solution evolves.

6.1.1 Viscous Terms

The first case is a flat plate compressible Blasius boundary layer flow with an inlet Mach number of $M_\infty = 2.0$ and a Reynolds number based on plate length of $Re_L = 10^4$. This case was computed with a 53×41 grid with an uniform axial spacing, a near-wall spacing of 10^{-3} blade lengths, and a uniform geometric stretching factor of 1.2 in the direction normal to the wall. Uniform supersonic freestream conditions are imposed at the upstream boundary, with a no-change condition at the exit. The upper surface uses a similiar radiative condition, while the wall surface uses the standard adiabatic no-slip condition, with the leading edge of the plate at the first grid point downstream of the inlet boundary. The number of points in the direction normal to the wall was chosen to insure that the Mach wave generated by the wall leading edge singularity intersects the exit boundary rather than the upper edge of the domain, as shown in Fig. 6.1. Figure 6.2 shows the velocity profiles and wall skin friction distribution along the plate. The theoretical curve is taken from the incompressible Blasius solution given by White [78, pp. 261-267], with a compressibility correction applied for the supersonic flow condition [78, pp. 586-591]. Agreement with the classical Blasius solution is quite satisfactory.

6.1.2 Turbulence Models

A flat plate in a supersonic flow with $M_\infty = 1.5$ and a Reynolds number based on plate length of $Re_L = 10^6$ was used to validate the one-equation turbulence model. This case is implemented with the supersonic inlet and mixed exit conditions described in Appendix B. Cases were run to show the effect of spacing normal to the wall on the computed boundary layer profile and skin friction. Three grids were used, each with 53 points spaced uniformly in the axial direction. The values of the first point spacing, normalized by the plate length, were 2.5×10^{-5}, 5.0×10^{-5}, and
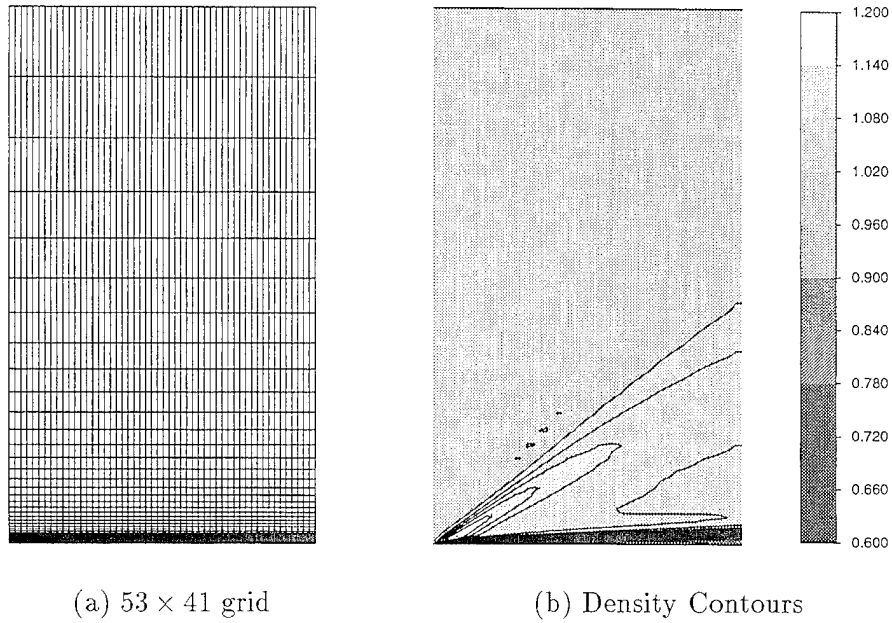
(a) 53 × 41 grid   (b) Density Contours

Figure 6.1: Blasius boundary layer solution.

1.0×10^{-4}, which is equivalent to $y^+ = 1.77$, 2.47, and 3.54 at the exit of the plate. In each case, a uniform stretching factor of 1.2 was applied in the normal direction, and the normal grid count was adjusted to keep the upper boundary approximately 1.5 plate lengths above the wall surface. Standard no-slip conditions are implemented along the surface and a freestream condition along the upper boundary. As described in Chapter 4, a freestream value of $\tilde{\nu} = \frac{\nu}{10}$ is used, which, using Eqn. 4.5, results in an imposed freestream value of $\nu_T = 2.8 \times 10^{-7} \nu$. Figures 6.3 and 6.4 show comparisons of the three cases in terms of nondimensional velocity profiles and skin friction. Both of the theoretical skin friction curves shown in Fig. 6.4 have been corrected for the effects of the $M_\infty = 1.5$ flow. The laminar skin friction curve uses the same correction as that described in the previous section for the Blasius boundary layer. The turbulent curve uses $C_f = 0.455 / \ln^2 0.06 Re_x$ [78, pg. 498] with the correction of van Driest [73] found in Schlichting [65, pg. 717]. These figures show that the
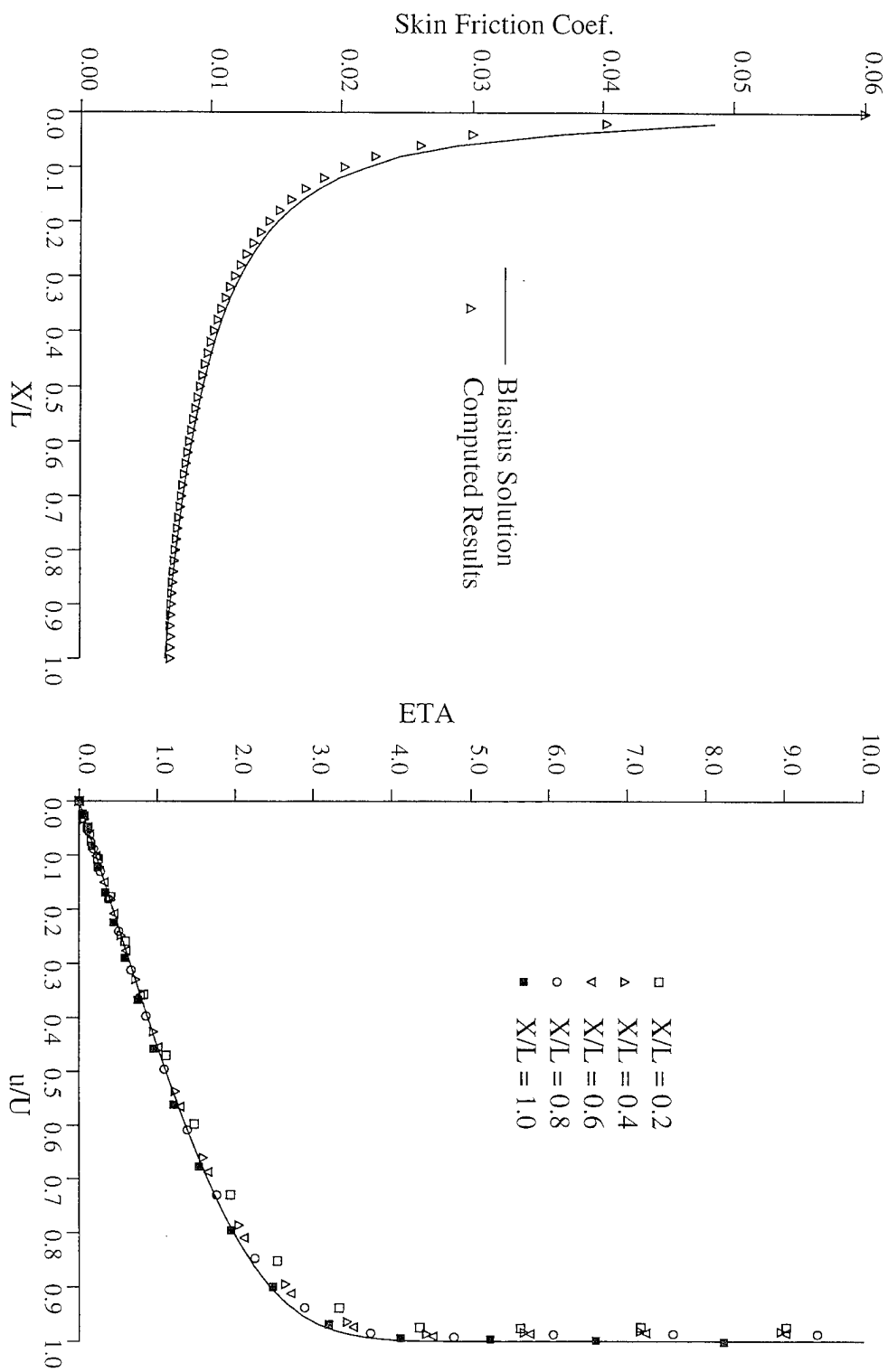
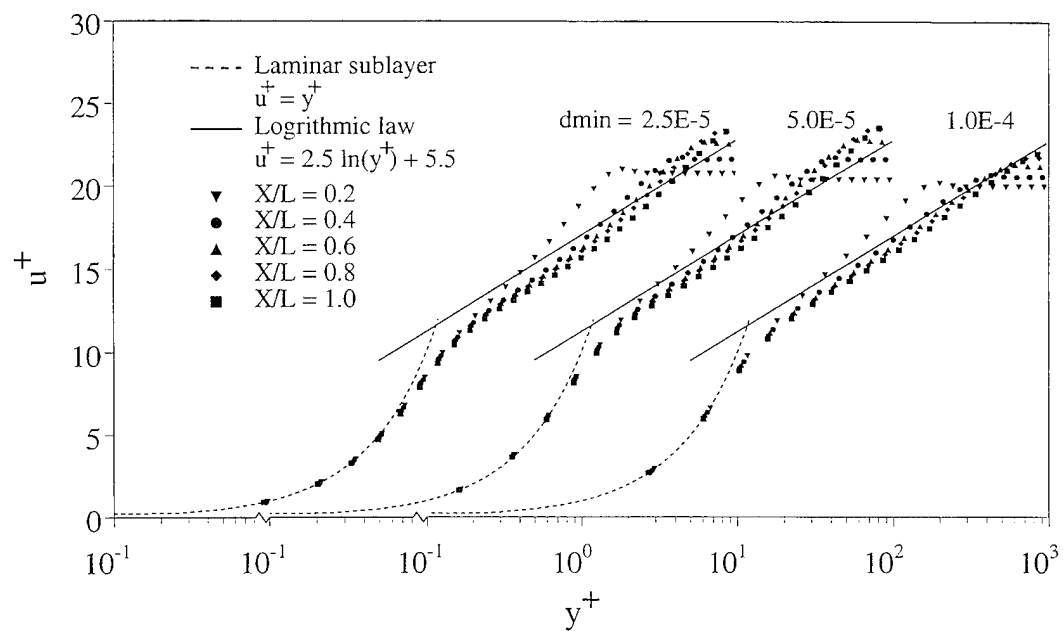Figure 6.2: Blasius boundary layer results for $M_\infty = 2.0$ and $Re_L = 10^4$.

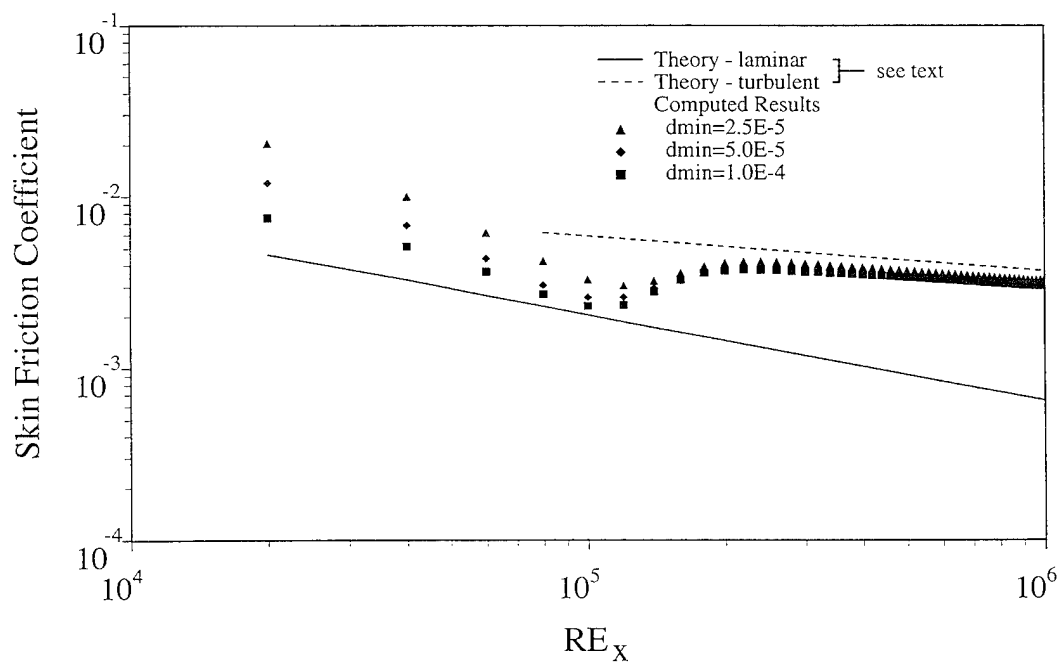Figure 6.3: Boundary layer profiles for three normal point spacings.

Figure 6.4: Skin friction distribution for three normal point spacings.

95

S-A model gives results which are quite acceptable, even for cases when there are only a few grid points within the laminar sublayer.

6.1.3 Unsteady Vortex Shedding

The time accuracy of the dual time stepping methodology was evaluated by a calculation of the well-known unsteady vortex motion behind a circular cylinder in a uniform crossflow. While a substantial amount of effort has been expended to compute this flow (see for example Beaudan [10]), the method described in this thesis lacks the temporal or spatial accuracy required to capture the full spectrum of temporal and spatial scales. Instead, the purpose of the test was to evaluate the ability of the method to capture the large scale motion through a prediction of the frequency of the vortex shedding behind the cylinder. A row of unit diameter cylinders spaced twenty diameters apart was studied using an inlet Mach number of 0.2 and Reynolds number based on diameter of 1000. For these conditions, the Strouhal number, as given by Schlichting [65, pp. 31-32], was expected to be 0.21. The multi-block O-H mesh used, shown in Fig. 6.5, contained a total of 4409 nodes in 265 patches. Starting with a uniform flow velocity condition throughout the domain, the computation tended to converge to a steady-state solution with a stable, symmetric vortex pattern. As the computation is continued, sufficient numerical error accumulates to make the solution asymmetrical, causing alternate vortex shedding to begin. In order to speed the formation of the vortex shedding, a surface blowing boundary condition was applied at five points along the upper surface of the cylinder for one shedding period. This condition produces sufficient solution asymmetry to trigger the shedding of the first vortex. After the blowing is discontinued, the alternate vortex shedding continues and rapidly becomes periodic.

One of the advantages of the dual time stepping method is the ability to select the temporal resolution of the unsteady solution through specification of the real time
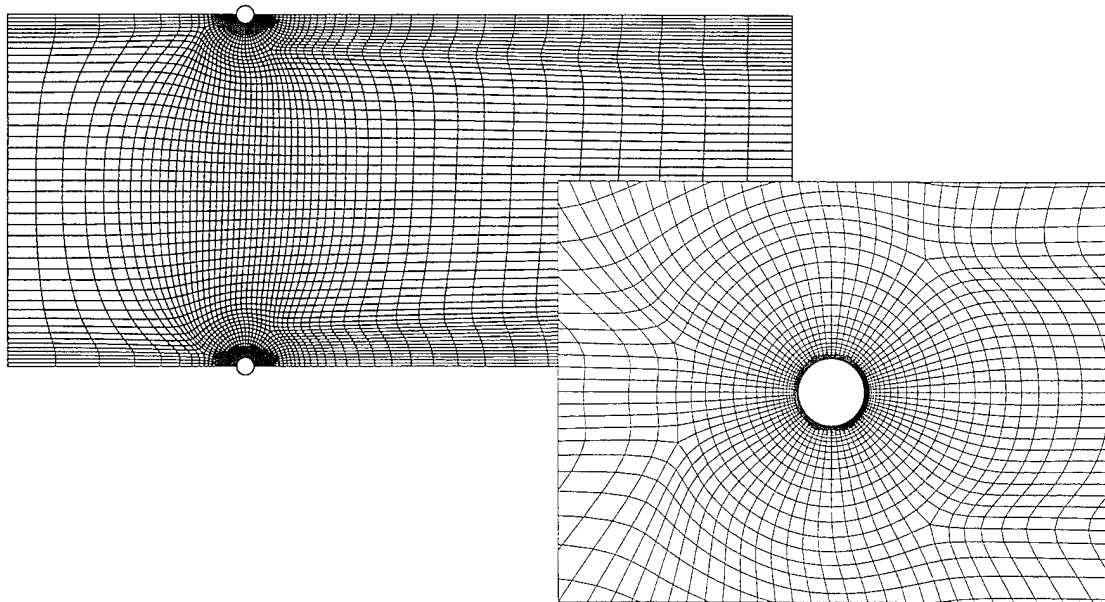
Figure 6.5: Grid used for cylinder vortex shedding test; detail of grid near surface showing multi-block structure.

step, Δt. This may allow the prediction of large scale unsteady phenomena, without requiring the resolution of the smallest time scales, which are related to the spatial resolution of the grid. The drawback, of course, is that the time step size required to yield an adequate level of temporal resolution is not known a priori. To illustrate this in the current test case, the period of the expected shedding was divided into 10, 20, 40, 80, and 160 equal real time steps. Because the initial transition from uniform to shedding flow was not of specific interest, the 20, 40, 80, and 160 time steps per period cases were started from the partially developed 10 step per period case at a nondimensional time of $t = 50$, each solution progressing until $t = 100$, by which time all solutions had become periodic. At each step in real time, the solution was converged in pseudo-time until the average of the four dependent variable residuals had been reduced by two orders-of-magnitude.

The vortex shedding motion is quite evident in Fig. 6.6, which shows Mach number contours for one period of the 80 iterations per period solution. The evolution of the unsteady lift and drag coefficients of the cylinder for the complete 80

iterations per period case are shown in Figs. 6.7 and 6.8. Experimental data of Roshko [63, 64] and Cantwell and Coles [13] indicate that the drag coefficient, for $Re_D = 1000$ flow, should be $C_D \approx 0.95 - 1.0$; the computed results yield an average value of $C_D \approx 1.05$, showing reasonable agreement. A comparison of all five iteration count cases is presented in Figs. 6.9 and 6.10, which clearly shows the considerable discrepancy in the computed shedding periods between the different solutions, an effect of the temporal resolution. These figures also show the asymptotic behavior of the solutions; as the iteration count is increased the solutions tend to converge to a final value. It is also apparent from these figures that there is a minimum number of iterations required to achieve even a marginally acceptable result; the 10 iterations per period case appears to be below this allowable level.

In order to evaluate the effect of grid resolution on the predicted shedding frequency, solutions were also obtained with a considerably finer grid. This fine grid was constructed by triggering adaptation of all patches in a coarse grid solution, forming a computational domain with 1060 patches containing 17297 nodes. The initial dependent variable values on the fine grid were found by interpolating from the same partially converged 10 iterations per period case used to start the coarse grid cases. Comparisons of the unsteady lift and drag coefficients for 40, 80, 160, and 320 iterations per period cases on the fine grid are given in Figs. 6.11 and 6.12. The solution on the finer grid shows a small increase in the peak lift and a higher overall drag coefficient. A examination of the velocity flow field downstream of the cylinder, as shown in Fig. 6.13, indicates the degree to which smaller scale vortices are captured on the finer grid.

Summaries of the results from all calculations on both grids are provided in Table 6.1, which also includes the results from the base grid case using a standard explicit time integration algorithm, i.e. without the dual time stepping technique. The explicit calculation on the refined grid was not completed due to the excessive

98

(a) $\frac{1}{4}$ period

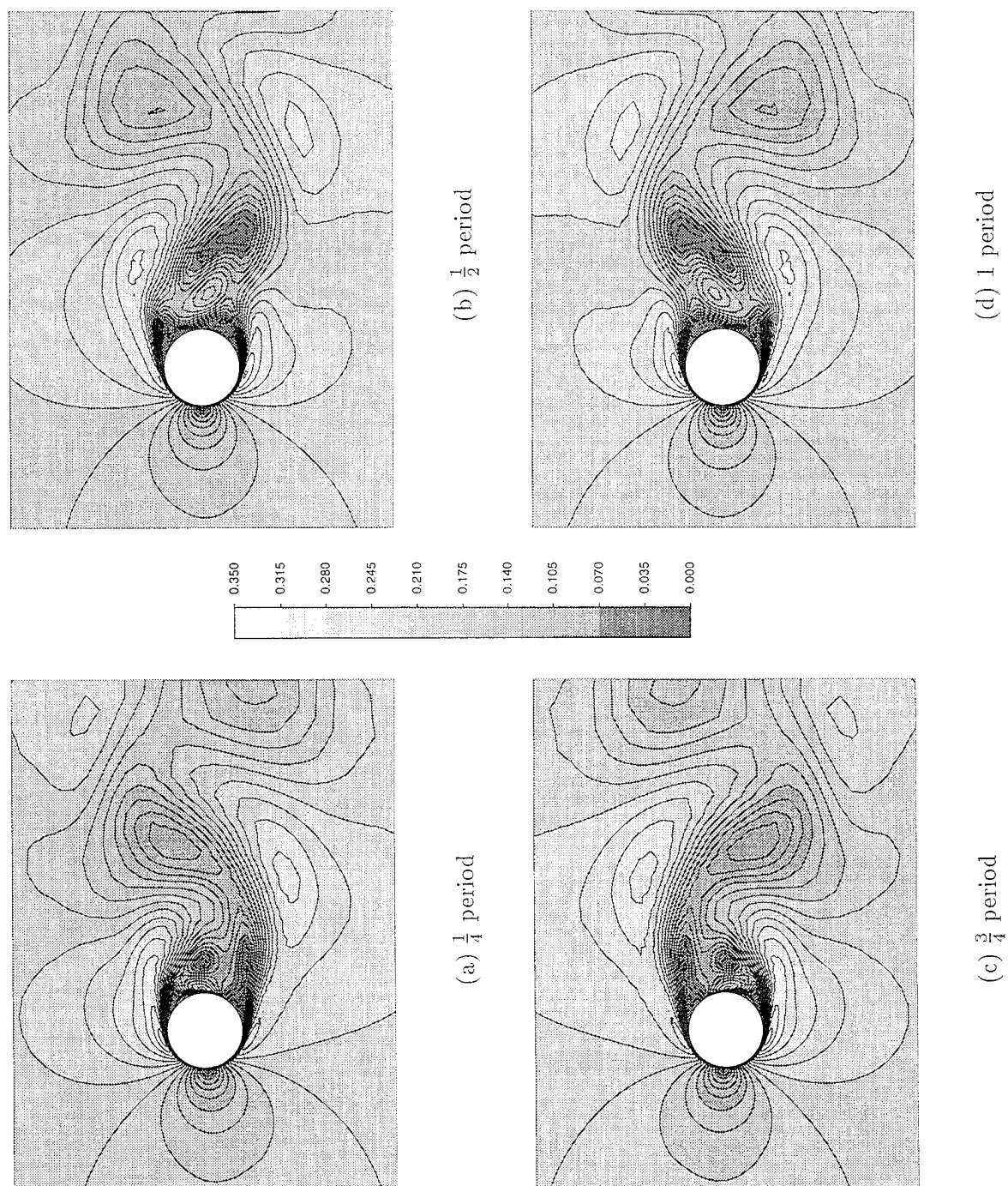(b) $\frac{1}{2}$ period

(c) $\frac{3}{4}$ period

(d) 1 period

Figure 6.6: Mach number contours over one shedding period; coarse grid solution with 80 iterations per period.
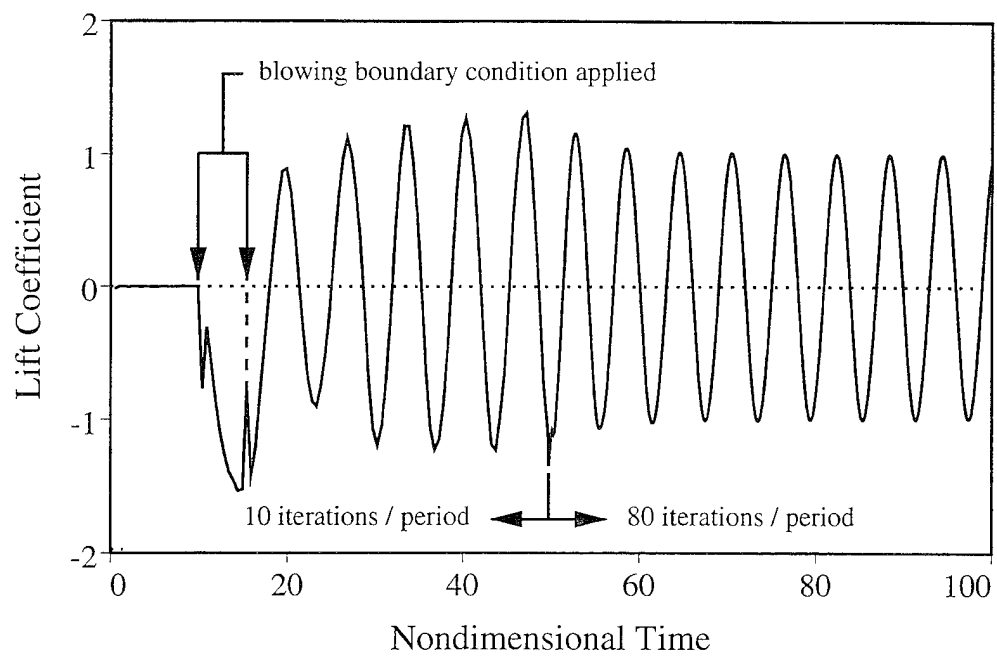
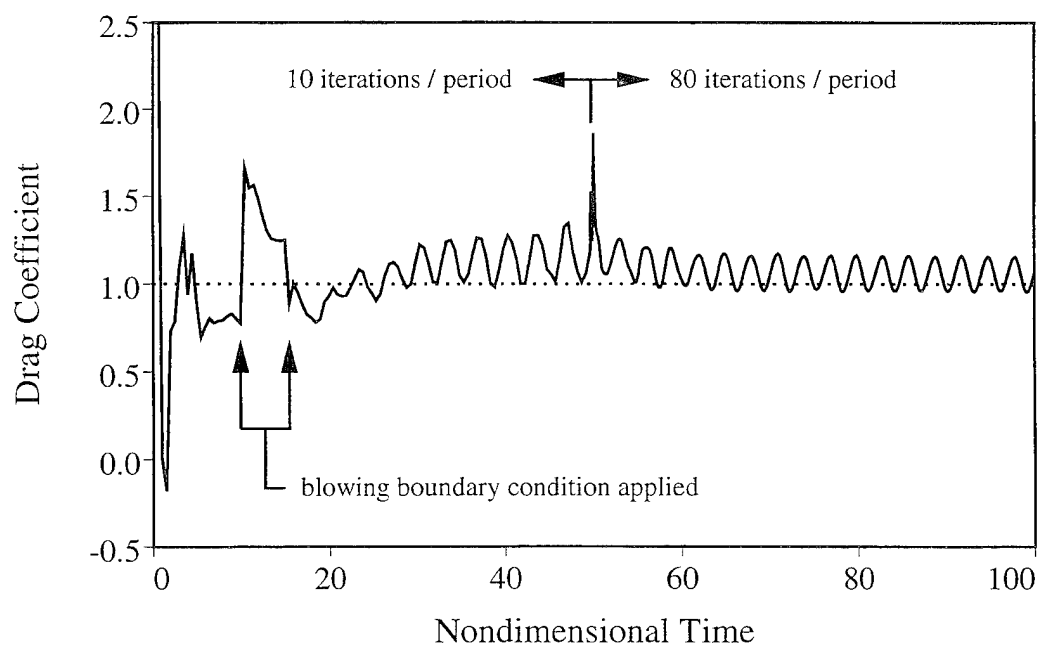Figure 6.7: Unsteady lift coefficient for cylinder vortex shedding; complete calculation.

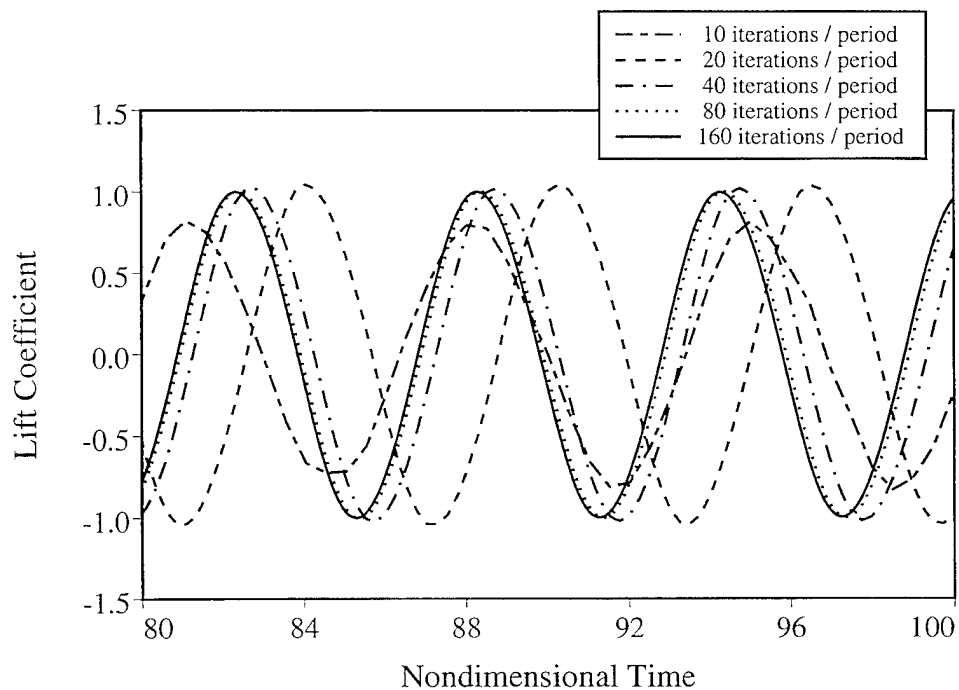Figure 6.8: Unsteady drag coefficient for cylinder vortex shedding; complete calculation.

100

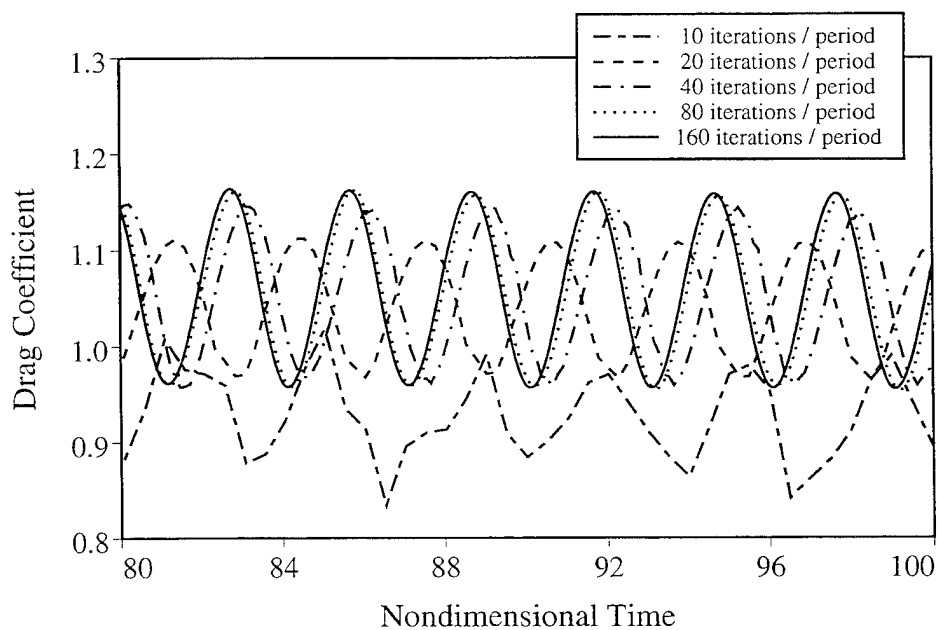Figure 6.9: Comparison of the unsteady lift coefficient using different iteration counts; base grid.

Figure 6.10: Comparison of the unsteady drag coefficient using different iteration counts; base grid.
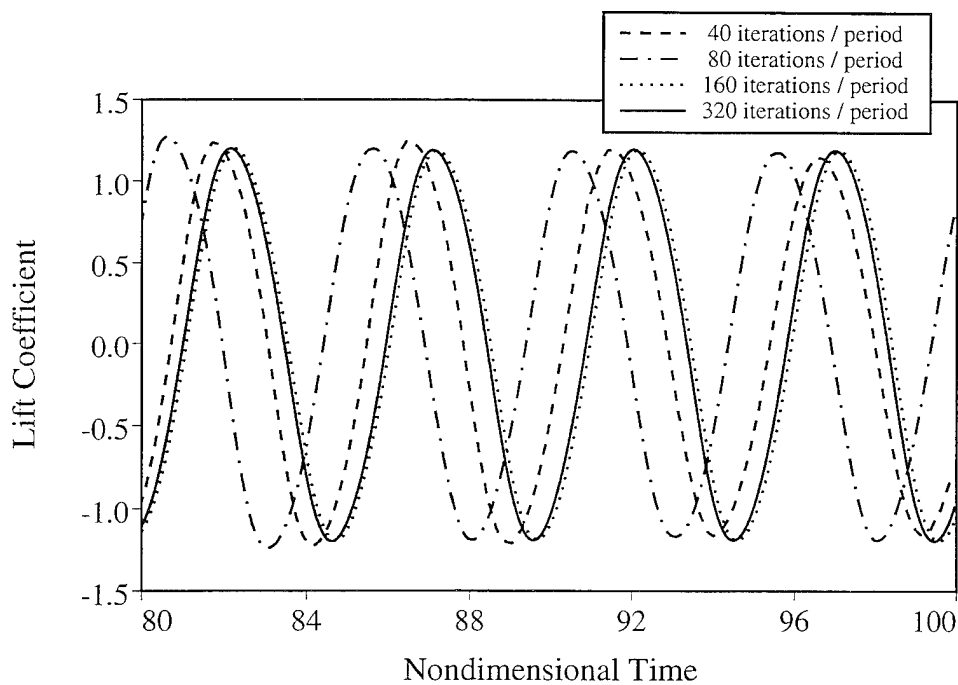
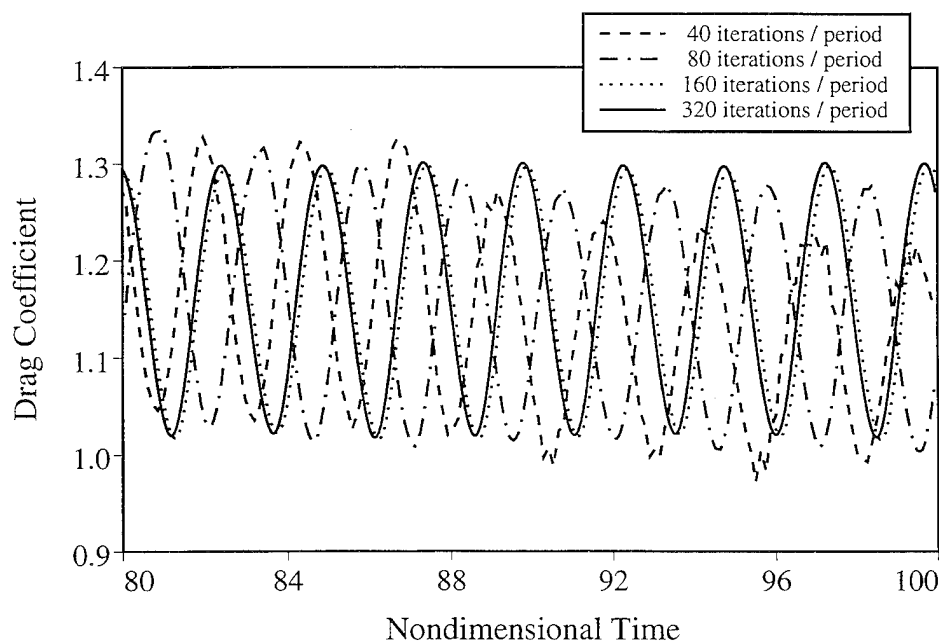Figure 6.11: Comparison of the unsteady lift coefficient using different iteration counts; refined grid.

Figure 6.12: Comparison of the unsteady drag coefficient using different iteration counts; refined grid.

(a) coarse grid solution - point A

(b) coarse grid solution - point B

(c) fine grid solution - point A

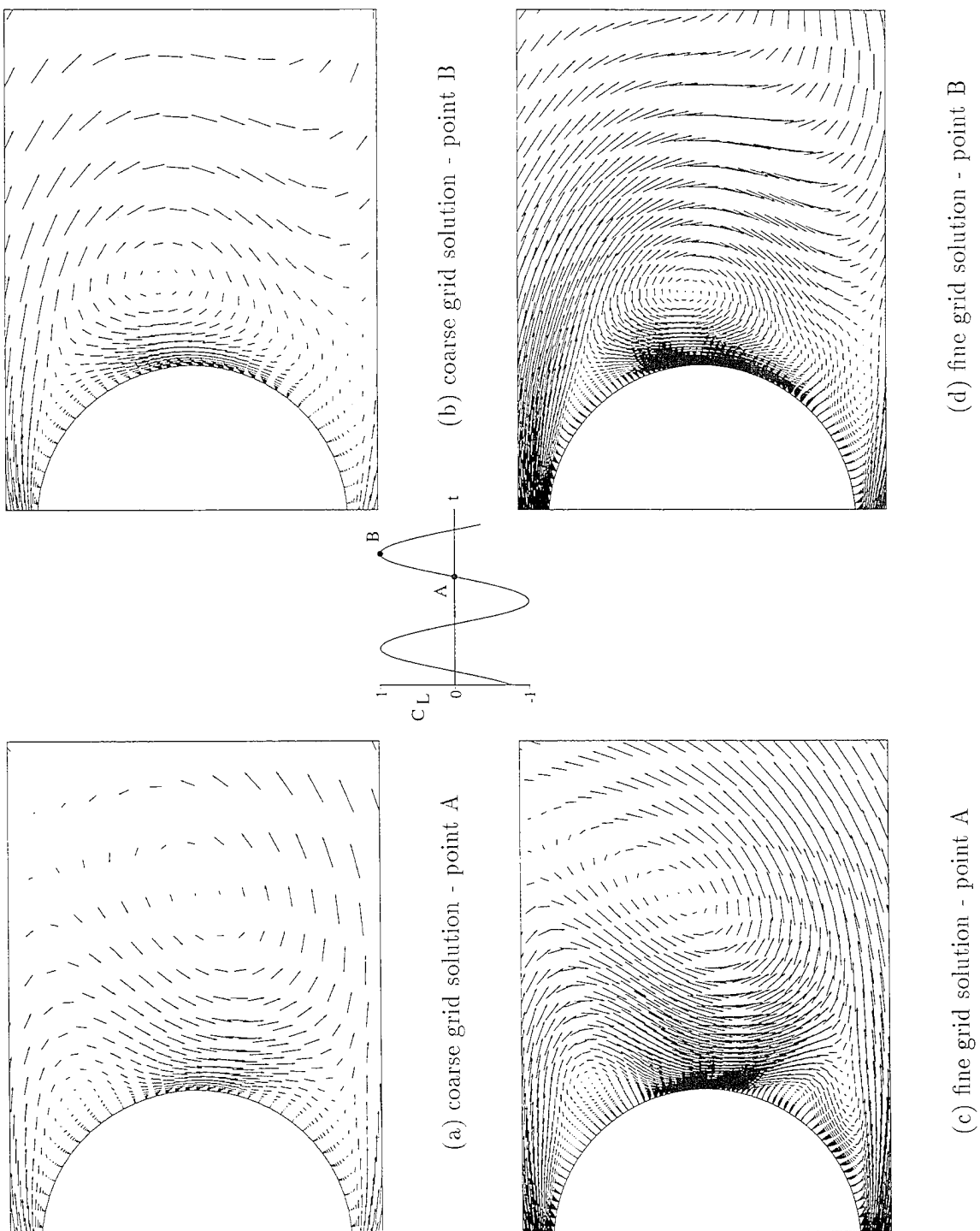(d) fine grid solution - point B

Figure 6.13: Coarse and fine grid velocity vectors at two points during a shedding period.

time requirement: an estimated 10 CPU hours per period. These results show that the computed Strouhal number for the coarse grid solution approaches 0.168, while the fine grid values approach 0.202, considerably closer to the expected experimental value of 0.21. This result seems quite reasonable considering the coarseness of the computational grid; additional levels of grid refinement would no doubt lead to further improvement in the computed Strouhal number. It may also be noted that fewer pseudo-time iterations are required to converge the solution at each real time step as the iteration count per period is increased. There is less change in the flow field structure with a smaller real time step, and therefore a more rapid convergence to the new state. Also included in the tables is an accounting of the effective CFL numbers used in the implicit scheme. These are computed as the ratio of the time step in real time to that in pseudo-time, with averages weighted by the cell areas. Finally, the CPU time requirements are provided for the single processor Silicon Graphics Power Indigo2 workstation used for all the calculations in this effort. These timing values clearly show the considerable time savings possible with the use of the implicit method.

Base Grid						
time steps per period	iterations per period	computed Strouhal No.	CFL Number max	avg	min	CPU time per period (min)
10	272	0.144	7833	330	8.0	2.7
20	568	0.160	3945	165	3.5	2.6
40	800	0.165	1961	82	1.8	3.9
80	1160	0.166	981	41	0.9	5.7
160	1646	0.167	492	21	0.4	7.3
expl	8514	0.164		3.5		203.4

Refined Grid						
time steps per period	iterations per period	computed Strouhal No.	CFL Number max	avg	min	CPU time per period (min)
40	2170	0.205	8736	223	3.5	47.4
80	3402	0.201	4390	111	1.8	63.0
160	4230	0.202	2170	56	0.9	96.6
320	5158	0.202	1066	28	0.4	122.4
expl	25720	-		3.5		614 (est)

Table 6.1: Summary of circular cylinder calculations.

6.1.4 Intra-Blade Row Interface

The interface between grids in relative motion was evaluated with a calculation of a hot streak in a uniform subsonic flow. The base flow has a uniform inlet condition with $M_\infty = 0.70$ and $Re_L = 10^3$, where L is the length of each grid block in the streamwise direction. The hot streak is modeled by increasing the inlet total temperature by a factor of 1.2 over a specified region. A similiar condition was used by Rai [55] for calculation of hot streaks in a turbine stage. The grids used, as shown in Fig. 6.14, were both 21×21 grids with uniform spacing in each direction. The downstream grid was translated vertically upwards with a speed equivalent to half of the inlet base velocity. Figure 6.15, showing temperature contours for a calculation without grid motion, shows no evidence of the interface. This is expected since the interface points are aligned and should not introduce any interpolation error. A similar steady calculation was also performed with the grids offset by half of one vertical cell spacing. Figure 6.16 shows the result of calculations using two and three points for the interpolation at the interface. It is evident that the two-point interpolation introduces errors at the interface not found when the higher-order interpolating function is used. Similar calculations were performed with and
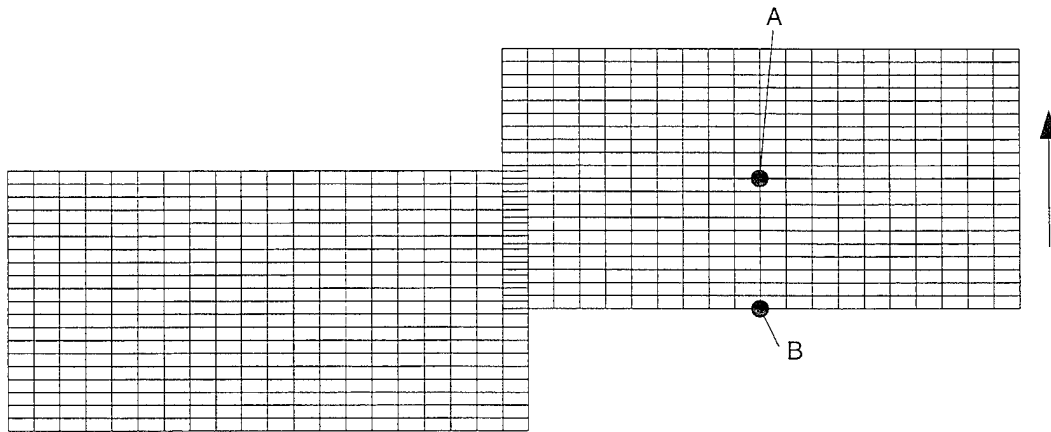

Figure 6.14: Coarse grids used for the interface test (21×21 points in each row).
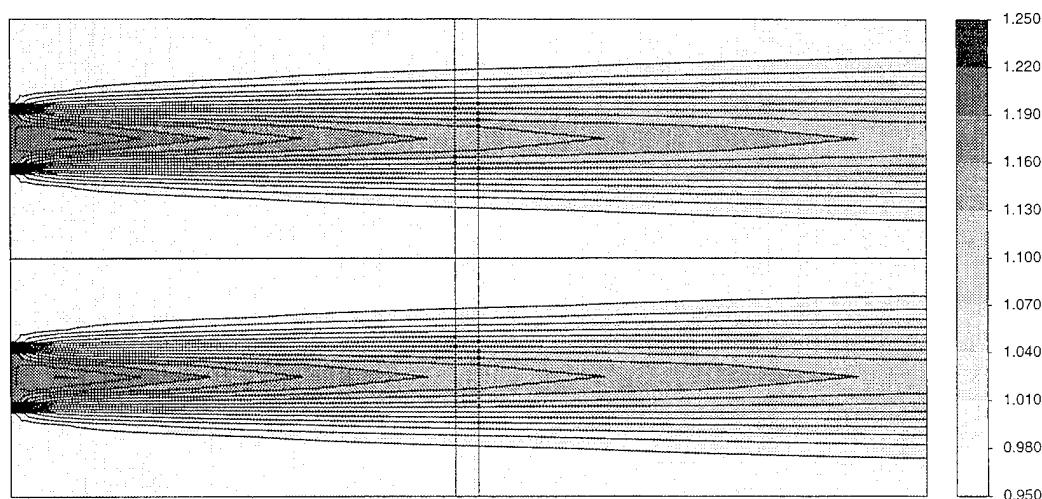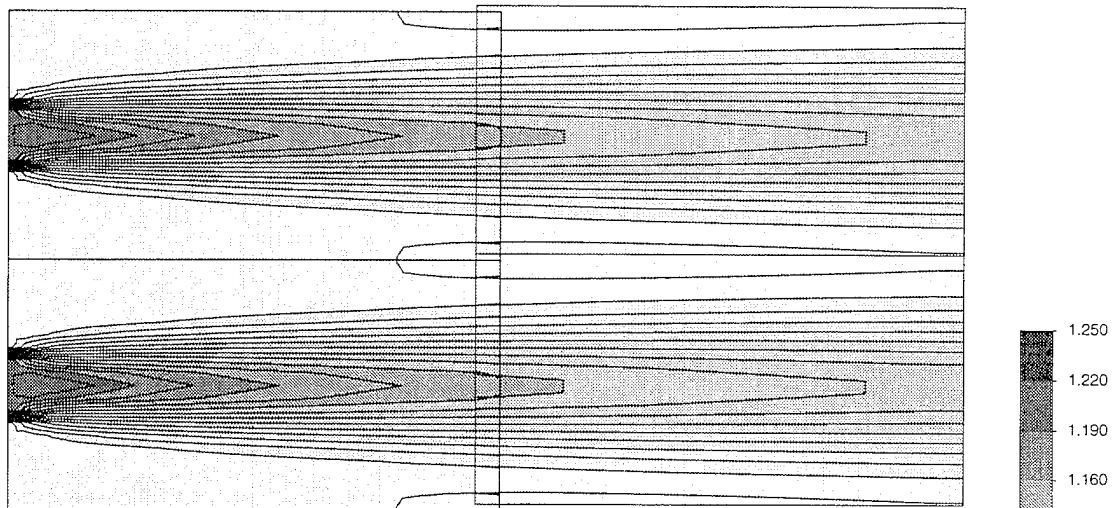
Figure 6.15: Temperature contours for the coarse grid solution without grid motion.

without conservation of the interpolated variables at the interface; no detectable difference in the solutions was found. A comment is in order about applying the conservative interface condition to variables with an expected mean value of zero. The conservative condition uses the ratio of the integrated flow variables, determined before and after the interpolation, to perform a correction of the mean value. For variables with a zero mean, this ratio should theoretically be a singularity, but in practice will be a ratio of two very small quantities. These numbers are both on the order of the roundoff error, and their ratio can therefore vary wildly. Logic has been introduced to detect this condition and add an arbitrary constant to both integrated values in order to avoid this singularity.

Unsteady calculations were performed on the same case in order to validate the rotational terms in the flow equations, as well as the performance of the interface in unsteady flows. As described in the preceding section, the required temporal resolution was determined by performing the calculation with a varying number of real time steps per grid passing period. The temperatures at points in the center of the downstream grid (denoted by A and B in Fig. 6.14) was monitored to determine periodicity. Using a time step equivalent to 20 iterations per period, the calculation

107

(a) 2 point interpolation
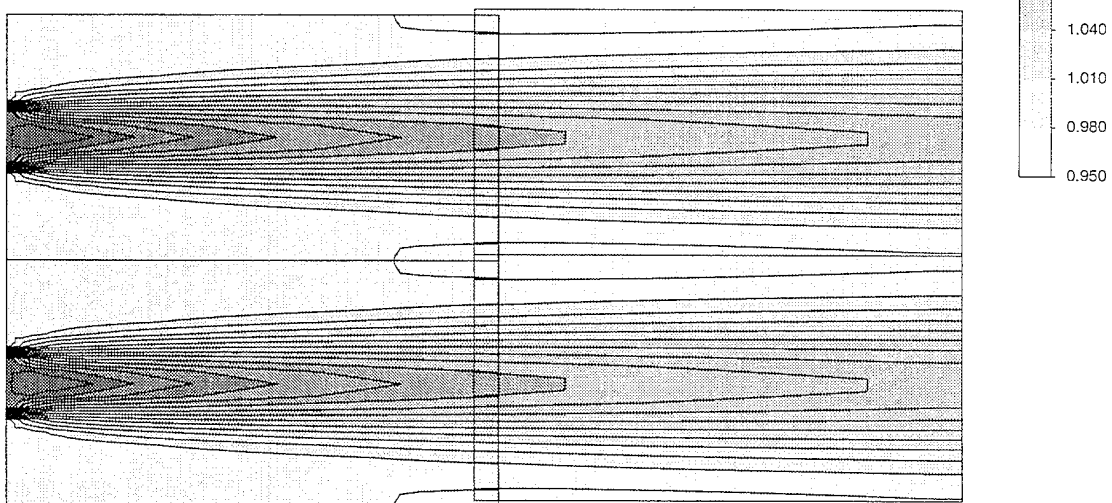
(b) 3 point interpolation

Figure 6.16: Temperature contours for solutions with different interface interpolation functions.
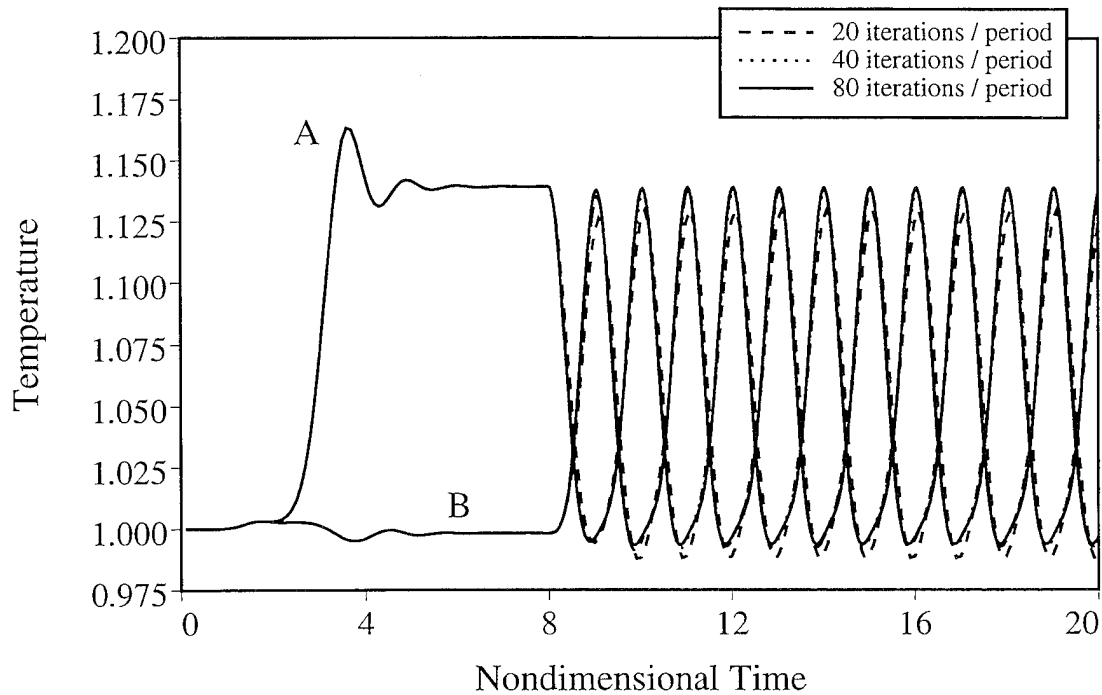
Figure 6.17: Temperature history for solutions with different iteration counts per grid passing period.

was first converged to a steady state solution without grid motion. Calculations were then run an additional six grid passings with 20, 40, and 80 iterations per passing period. The 20 iteration case shows a discrepancy between the steady state temperatures and the peak temperatures in the unsteady case; this behavior is absent from the 40 and 80 iteration cases. Figure 6.18 shows the temperature contours at the end of the six grid cycles. The 20 iteration case shows a distortion of the hot streak in the moving downstream grid, which is improved as the iteration count per period is increased. The steadiness of the profile is particularly good considering the coarseness of the grids.

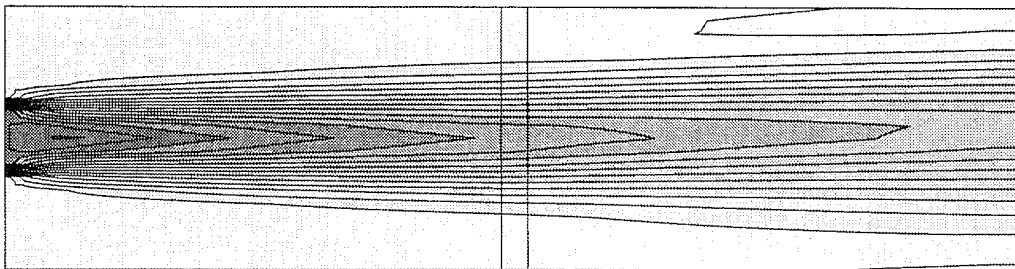### 6.1.5 Adaptation to Unsteady Flows

A test of the unsteady adaptation was performed for the hot streak case described above. At each step in real time, the grid was adapted to undivided differences in temperature after the average L^2 norm of the solution residuals had fallen a
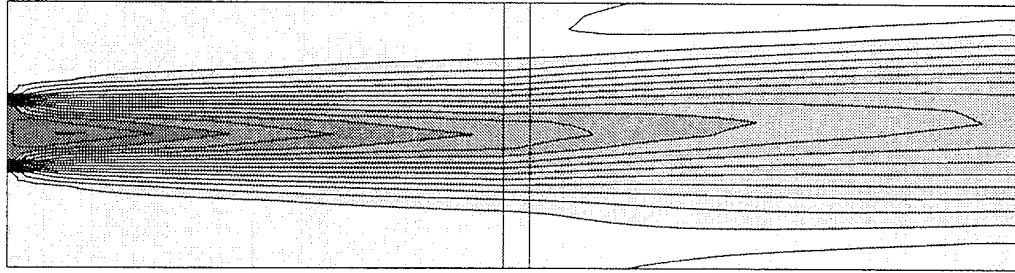
(a) 20 iterations / period
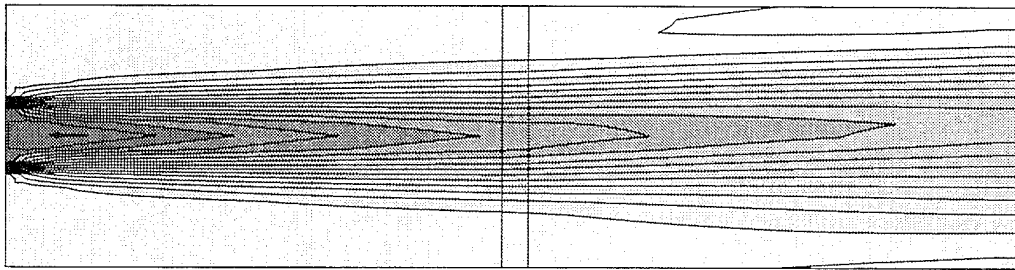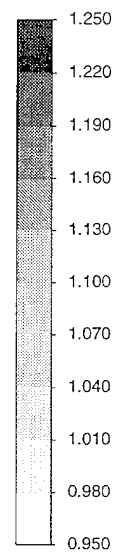
(b) 40 iterations / period

(c) 80 iterations / period

Figure 6.18: Comparison of temperature contours for coarse grid solutions with different iteration counts per grid passing period.

single order of magnitude. The solution was then allowed to converge another three orders-of-magnitude, based on the solution residual immediately following the adaptation. In general, four iterations in pseudo-time were performed before the grid was adapted, followed by another seven to complete the real time iteration step. Control over the total number of grid points was maintained by allowing only two levels of cell refinement beyond the base grid. An additional solution using a globally fine grid was also obtained in order to compare to the adapted solution. Although the number of points changed at each iteration, the adapted grids typically contained approximately 460 patches and 7800 nodes as opposed to the fine grid of 800 patches and over 13000 nodes.

There are a number of differences to note between the original coarse grid solution and that obtained using the medium, fine, and adapted grids. The first is that the enhanced grid resolution lessens the numerical dissipation and consequently reduces the spreading of the hot streak. This is illustrated in Fig. 6.19 which shows the comparison of the coarse, medium, and fine grid solutions. This improved resolution is also reflected in the change in the peak temperature shown in Fig. 6.20, which includes an overlay of the fine grid steady solution values. Figure 6.20 also shows that the adapted solution includes a small dip or "trough" in the temperature distribution on one side of the streak. An attempt was made to remove this anomaly by increasing the number of iterations per grid passing. Figure 6.21 shows a close-up of the temperature distribution for 40, 80, and 160 iterations per grid passing overlaid with the fine grid values. Increasing the number of time steps does reduce the size of the dip, but it is equally possible that this is an artifact of the adaptation itself; this phenomenon currently remains unexplained.

Figure 6.23 shows a selected number of different images of the grid at various times in a grid passing period. Improvement in the resolution of the streak is evident over that shown in the previous images without the adaptation. Slight distortion of

111

the contours is seen in the moving grid region, and is likely due to "noise" in the location of the edge of the region of finest cells. Some improvement in the solution can be made by reducing the cell division threshold, but this results in the finest level of adaptation in all patches in the downstream grid, defeating the purpose of the adaptive algorithm. There is also some evidence of the error introduced by the hanging node interfaces, particularly where an internal or external corner exists between adaptation levels. These corners cause an accumulation of stretching error in both directions, resulting in a "worst case" test for the interface algorithm.
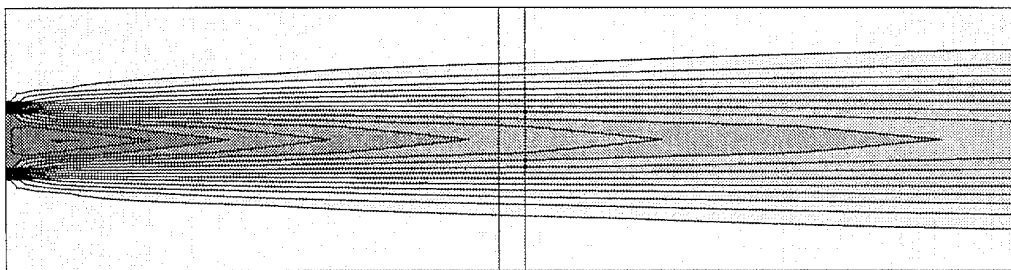
(a) Coarse grid solution (two 21 × 21 grids)
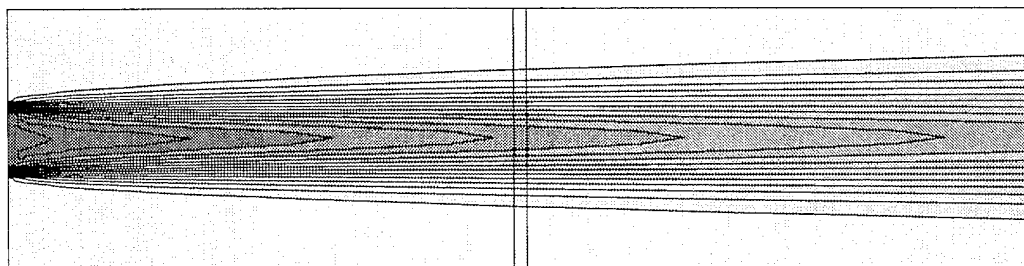
(b) Medium grid solution (two 41 × 41 grids)
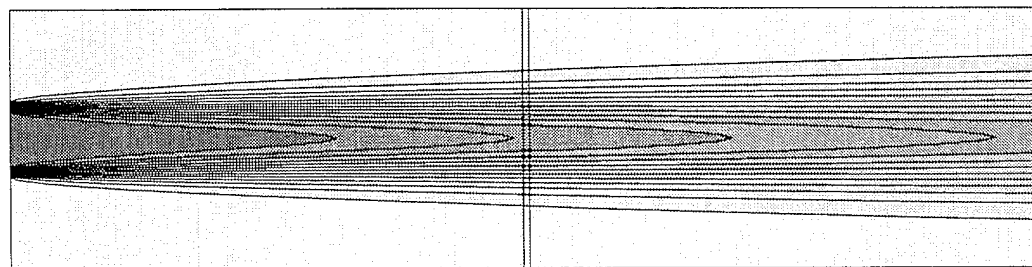
(c) Fine grid solution (two 81 × 81 grids)
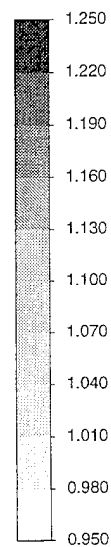
Figure 6.19: Comparison of coarse, medium, and fine grid temperature contours for solutions without grid motion.

113

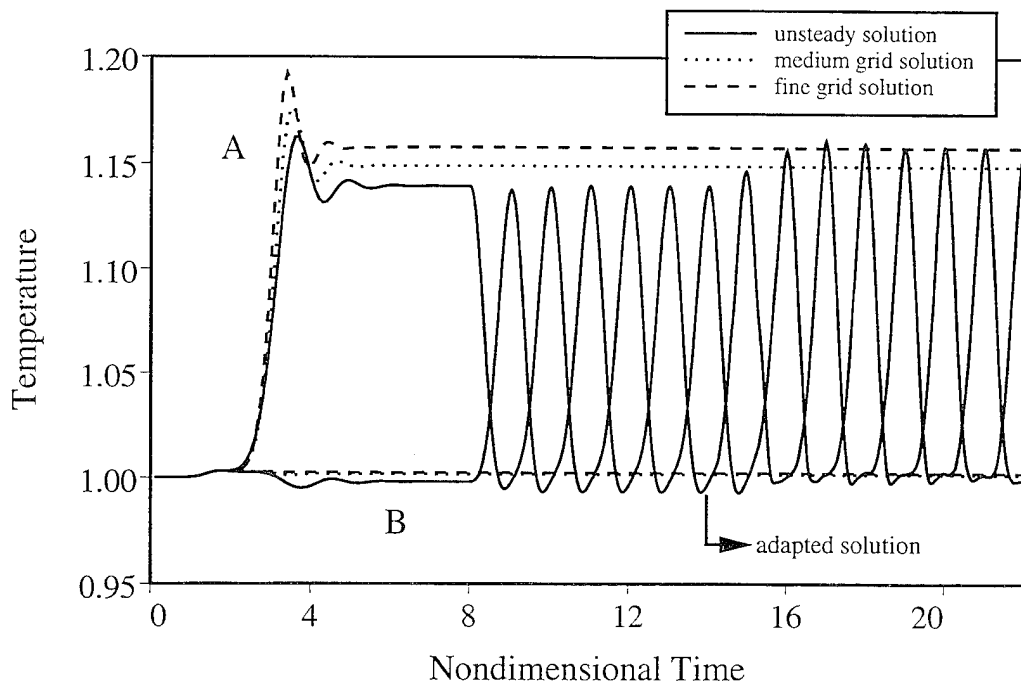Figure 6.20: Temperature history for fine, medium, and adapted grid solutions.
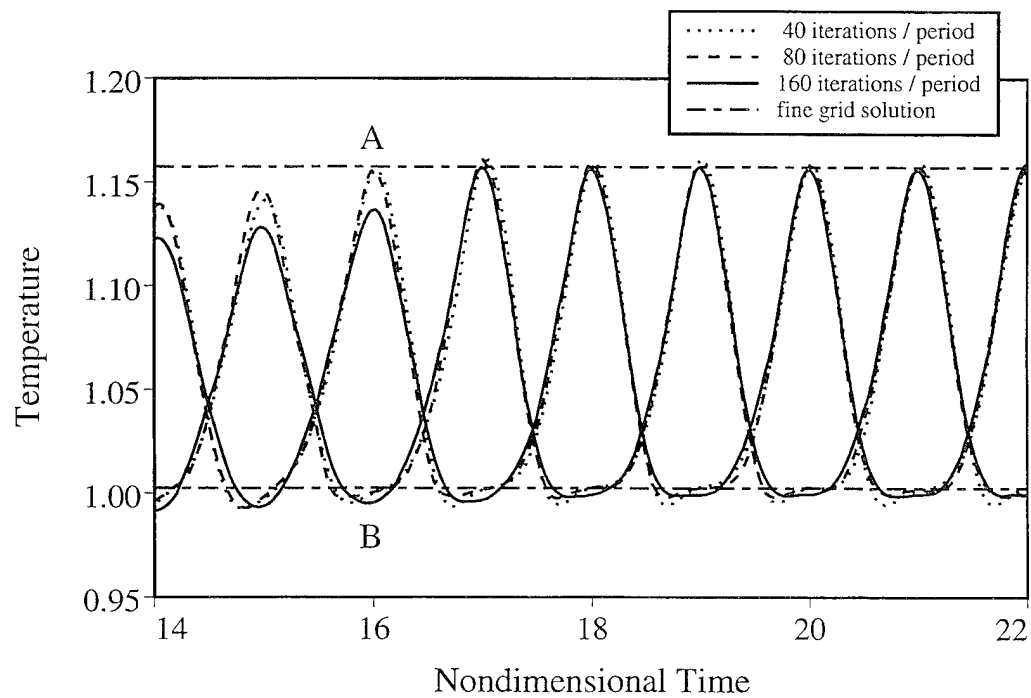
Figure 6.21: Comparison of unsteady adaptation solutions.
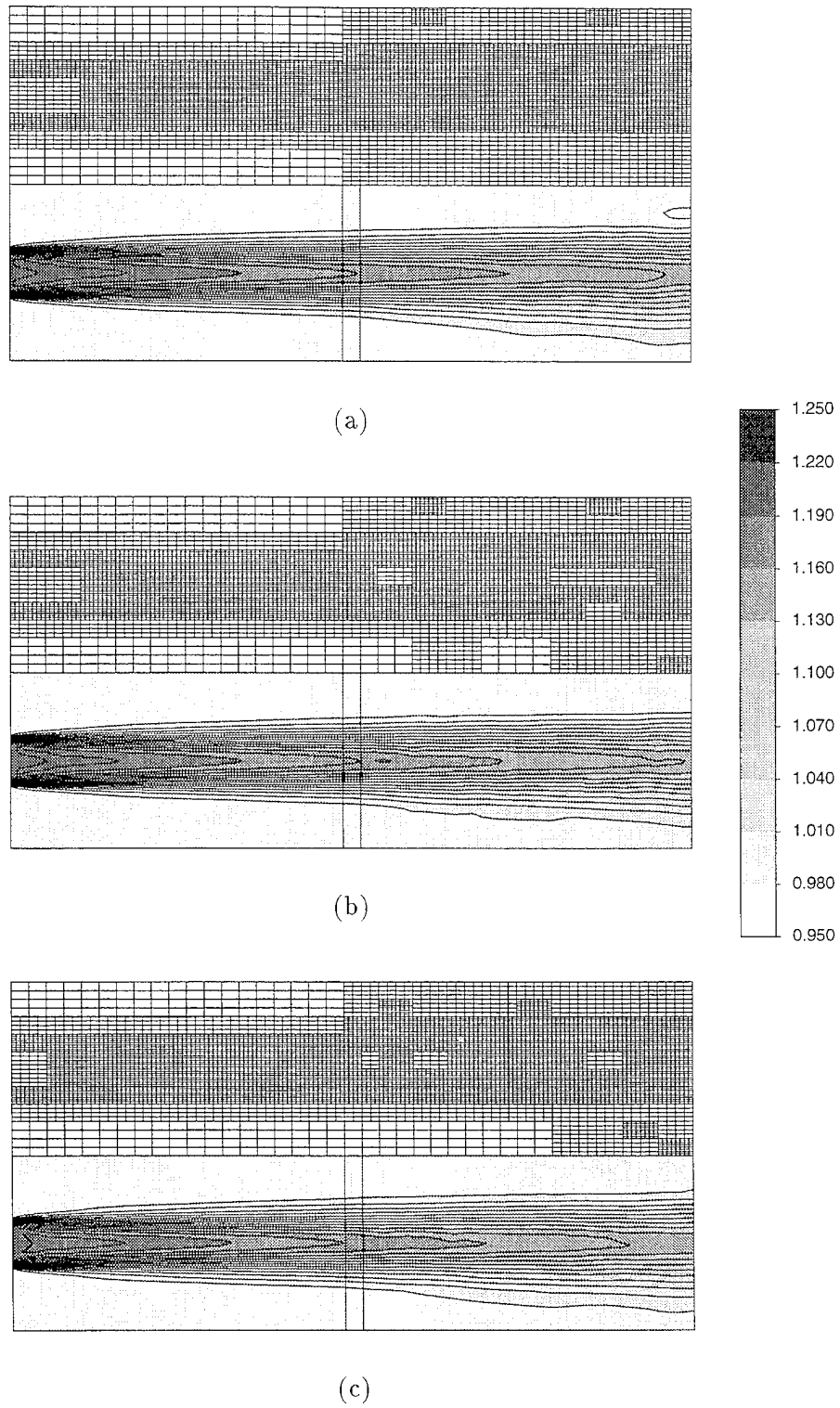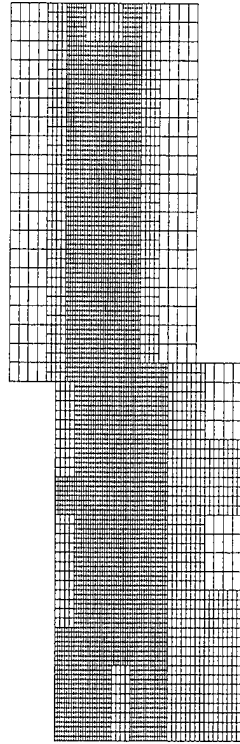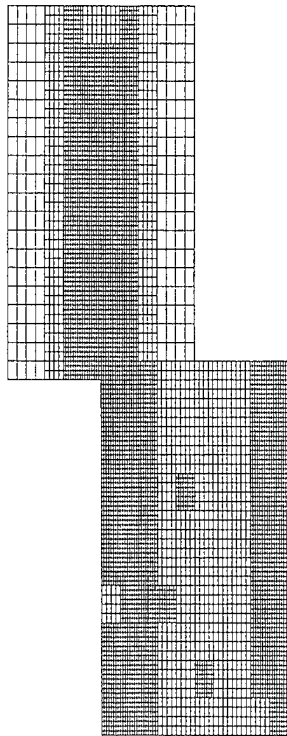
114

(a)

(b)

(c)

Figure 6.22: Comparison of adapted grids and temperature contours for three consecutive periods.
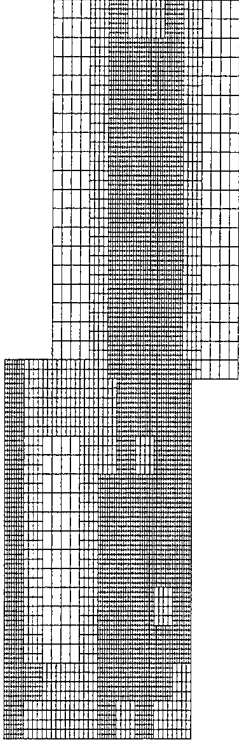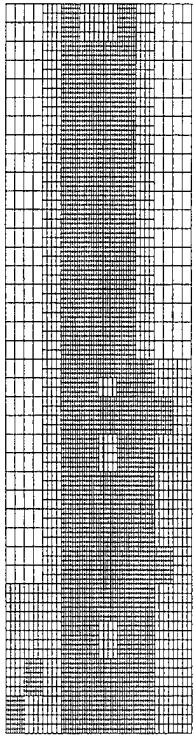
115

(a) $\frac{1}{4}$ period

(b) $\frac{1}{2}$ period

(c) $\frac{3}{4}$ period

(d) 1 period

Figure 6.23: Adapted grid during period (downstream grid moving upwards).
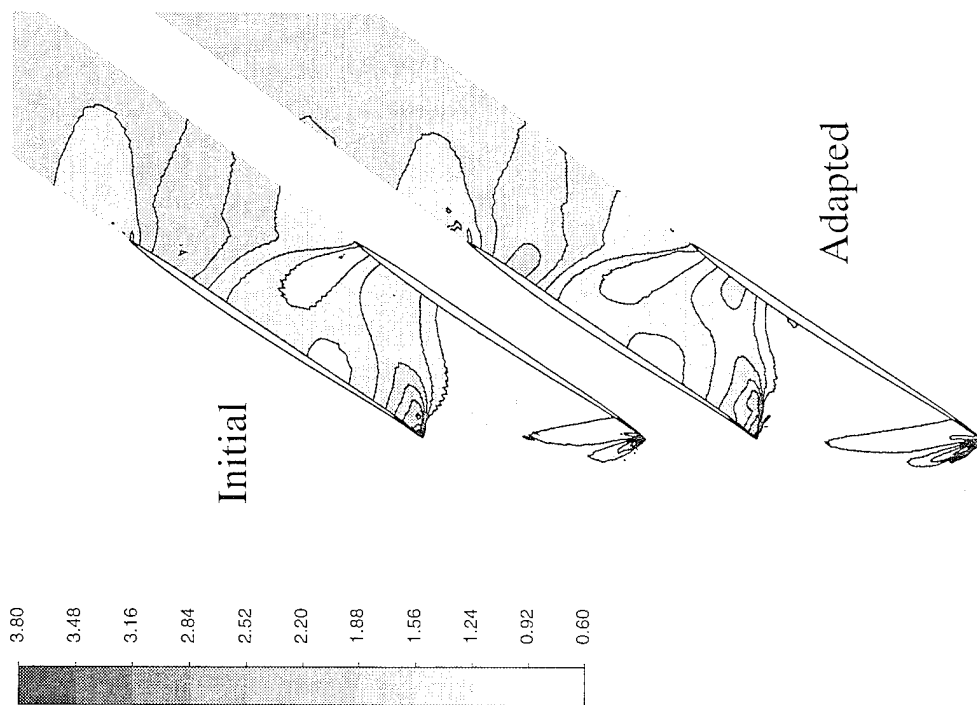
6.1.6   Isolated Cascade Cases

Two single blade row cases are presented in order to validate the computational model against available experimental data. Each of these cases are representative of typical designs found in modern transonic turbomachinery.

TRANSONIC FAN CASCADE

The compressor cascade used is a near-tip section from a transonic fan design with experimental data for the surface pressure distribution available in Fleeter et al. [25]. The case chosen has an inlet relative Mach number of 1.535 and a Reynolds number based on blade chord and inlet relative conditions of 1.16×10^6; experimental data are available for a static pressure ratio range of 1.22 - 2.30.

Calculations were performed for a back pressure ratio of 1.505 with the experimentally determined streamtube contraction (defined as the ratio of the exit-to-inlet streamtube thickness) of approximately 96%, and with values 4% above and below this value; the streamtube radius was held constant. In each case the computations proceed on an initial grid until the positions of the bow and passage shocks have stabilized. With the low resolution of the typical initial grid, the passage shock can be smeared over approximately 20% of the blade surface. The initial adaptation cycles are performed with fairly low gradient thresholds, serving to stabilize the shock position and reduce the smearing of the normal shock. The change in shock position and definition after two adaptation cycles may be seen by comparing the original and the adapted grids and flow fields given in Fig. 6.24.

A comparison of the three computed blade surface pressure distributions to that measured by Fleeter, et al. [25] is seen in Fig 6.25. Good agreement with experiment is obtained at the design contraction ratio, and a reduction in the passage shock strength with decreased contraction ratio is found as expected. It is

3.80
3.48
3.16
2.84
2.52
2.20
1.88
1.56
1.24
0.92
0.60

Initial

Adapted

(b) static pressure

Initial

Adapted

(a) patch boundaries
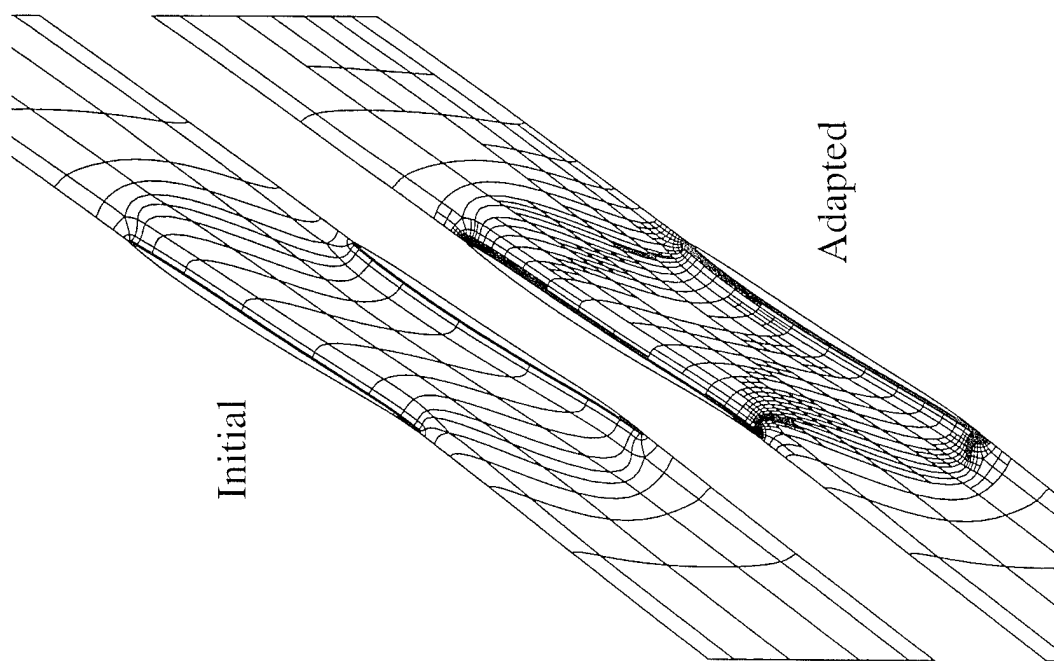
Figure 6.24: See caption page 119.

118

x E-5

4.50
4.05
3.60
3.15
2.70
2.25
1.80
1.35
0.90
0.45
0.00

Initial

Adapted

(d) turbulent eddy viscosity

1.80
1.62
1.44
1.26
1.08
0.90
0.72
0.54
0.36
0.18
0.00
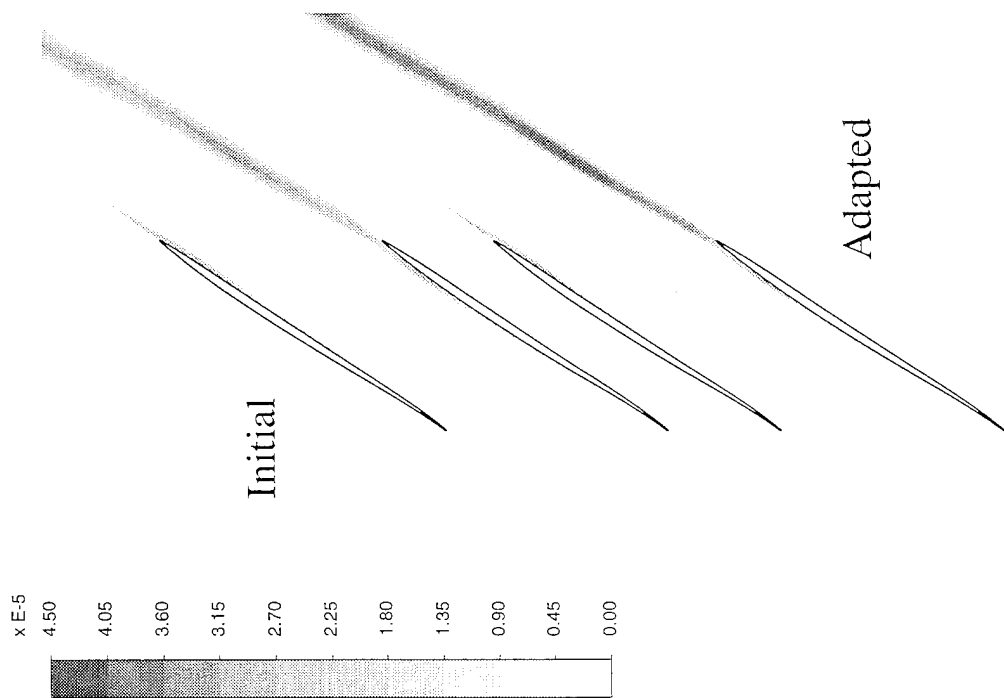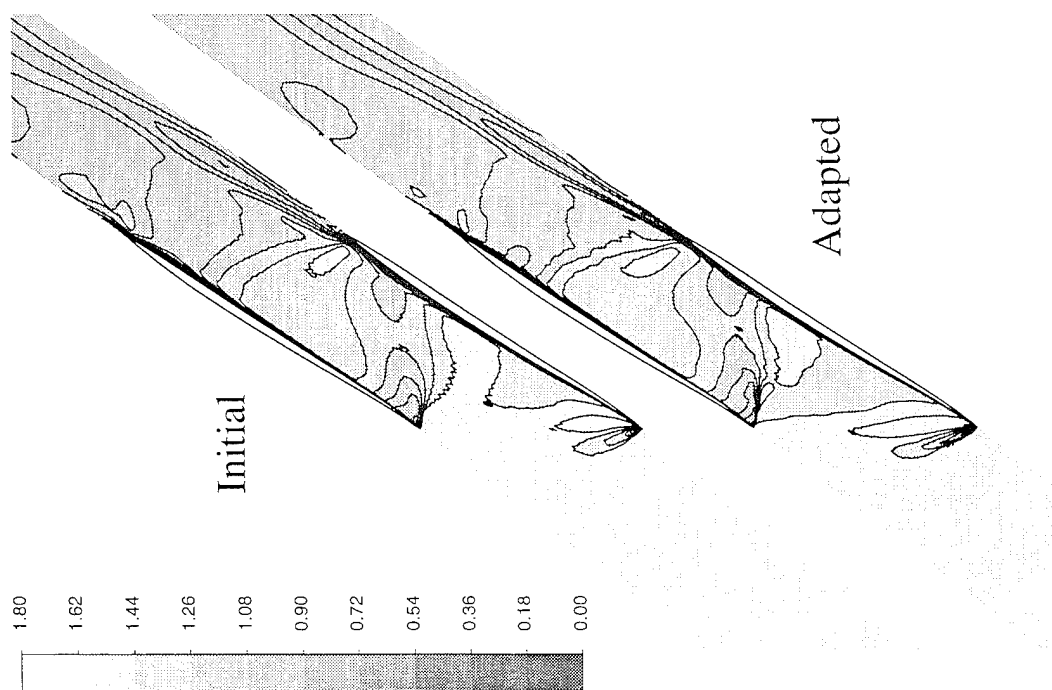
Initial

Adapted

(c) Mach number

Figure 6.24: Fan cascade initial and adapted solutions for $M_\infty = 1.535$, $PR = 1.505$.
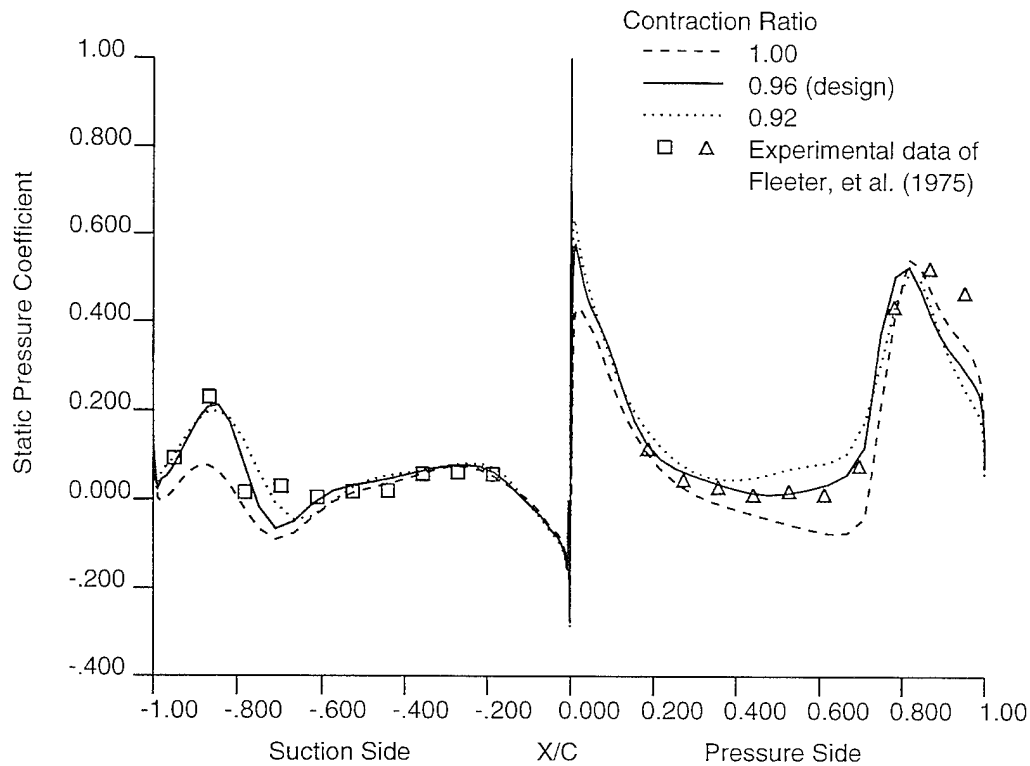
Figure 6.25: Comparison of surface pressure distribution for three streamtube contraction ratios.

interesting to note that since the overall cascade pressure ratio is fixed, the pressure rise is redistributed within the cascade depending upon the specified contraction. Therefore a cascade with a lower contraction ratio will have a stronger passage shock and a weaker leading edge bow shock. Comparison of the total pressure ratio distribution in the exit wake at a traverse location 25% chord downstream of the trailing edge is given in Fig. 6.26. Best agreement with experiment is found for the design contraction ratio both for wake shape and for the mass averaged losses. The influence of contraction ratio on the passage shock strength is also seen here in the shift in position of the wake centerline indicating the variation in turning between shocks of different strengths.
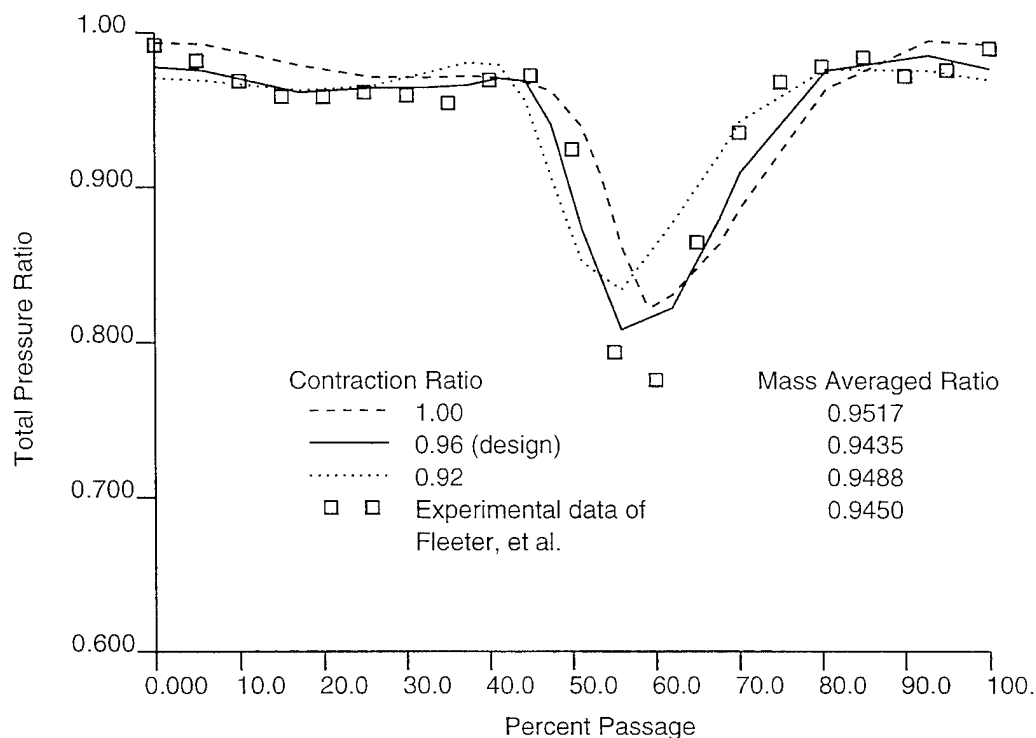
Figure 6.26: Experimental and computed total pressure ratio 25% chord downstream of trailing edge.

TRANSONIC TURBINE ROTOR

Computations are also compared with experimental results for the transonic turbine rotor cascade of Ashworth et al. [6]. The comparison case is the nominal design point, with an inlet Mach number of 0.59, an incidence angle of 58.06 degrees, a Reynolds number of 9.19×10^5 based on the blade chord, and an isentropic exit Mach number of 1.18. The calculations include the test cascade expansion of approximately 12% and use an isothermal wall specification with $To/T_{wall} = 1.5$. The adaptation cycle used is similiar to that described for the compressor flows above. The initial and adapted grids and flow fields are presented in Fig. 6.27, clearly showing the improvement in the prediction of the blade boundary layer and wake over that in the initial grid solution. The comparison with the experimentally measured blade loading in Fig. 6.28 shows a reasonable agreement with the available data. The

experimental and computed heat transfer distributions are given in Fig. 6.29, with the Nusselt number defined as [6]:

$$Nu = \frac{\dot{q}_w c_\theta}{(T_o - T_w)k}$$

(6.1)

where $\dot{q}_w$ is the heat transfer at the wall and c_θ is the tangential chord. Figure 6.29 highlights the improvement in the predicted values as the resolution of the blade boundary layer is improved.
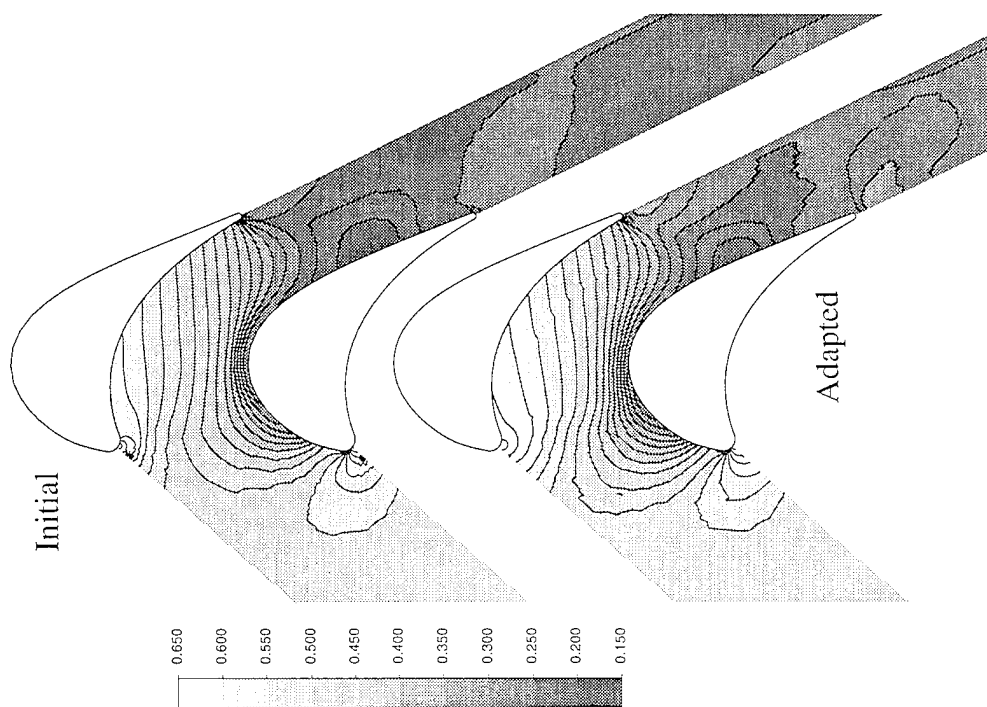
(b) static pressure
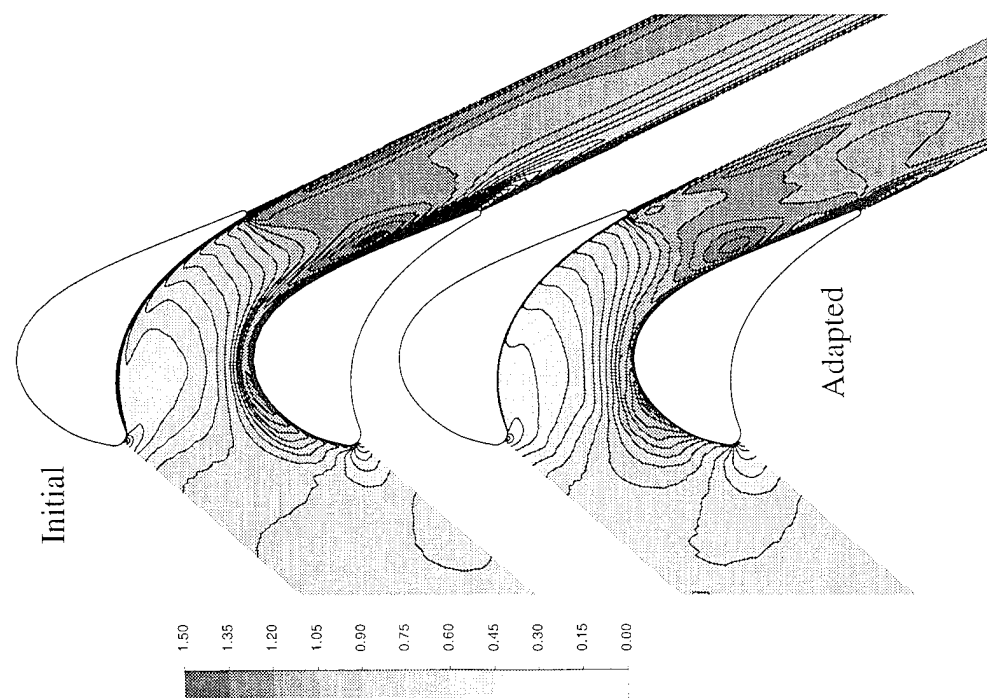
(a) Mach number

Figure 6.27: Turbine rotor initial and adapted solutions with design back pressure.
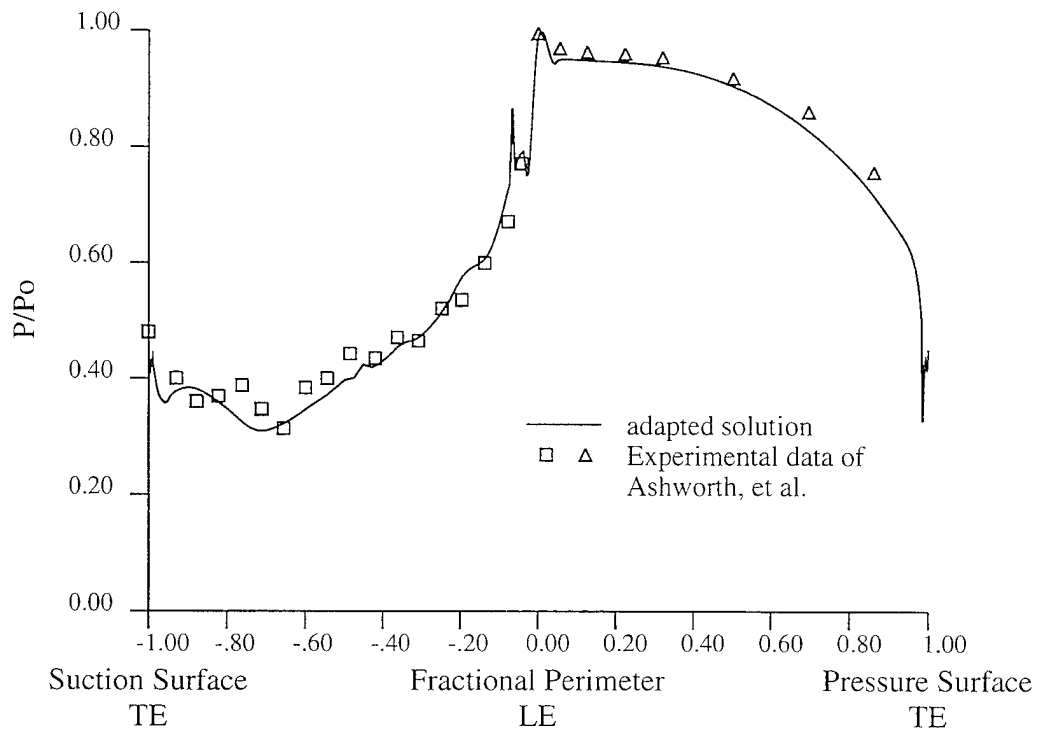
Figure 6.28: Turbine rotor blade loading diagram for the adapted solution.
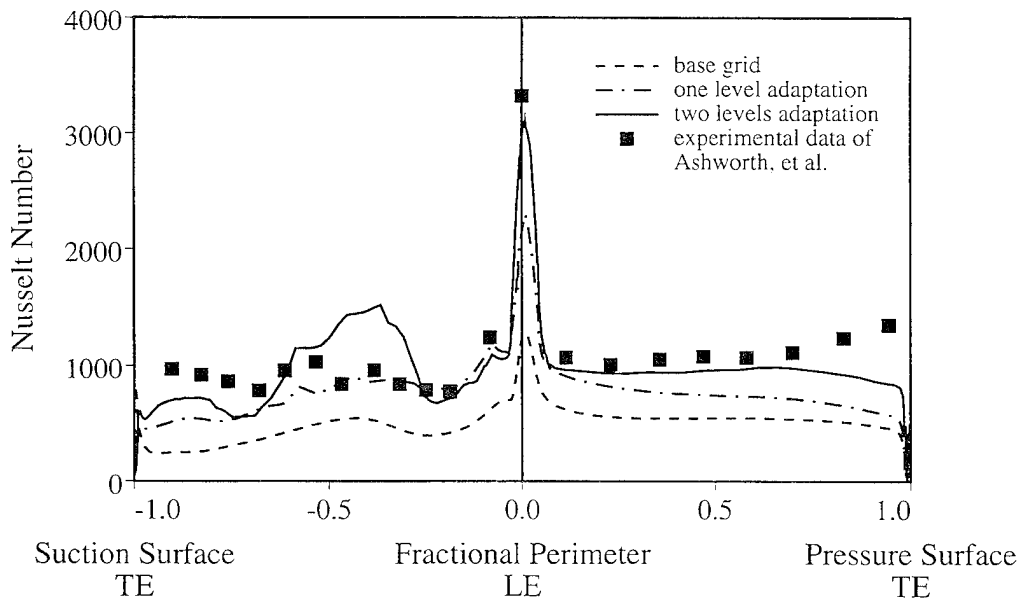
Figure 6.29: Turbine rotor heat transfer distribution for the adapted solution.

124

6.2 Turbine Stage Application

In order to demonstrate the complete computational model, calculations of the unsteady interaction in a transonic turbine stage are presented. This turbine is a single stage design with 36 guide vanes and 61 rotors blades; the rotor is the same as that presented in the previous section. Design parameters for the full scale machine include a pressure ratio of 4.3 across the stage, a vane isentropic exit Mach number of 1.18, and a wall/gas temperature ratio of 0.63. The Reynolds number based on the guide vane true chord and isentropic vane exit conditions is $Re = 2.47 \times 10^6$. The actual machine uses film cooling in both the vane and rotor rows which is not included here. The section modeled is at a near mid-span location and includes an assumed radius and streamtube thickness change taken from the stage endwall geometry and previous calculations on the stage [1]. Although the computer code described here has been written in a general form, allowing any number of blade rows to be calculated, each with an integer number of passages, a 1:1 blade ratio grid has been used to reduce the overall computational requirements. To maintain the original blade solidity (ratio of blade chord to cirumferential spacing), the downstream blade row has been uniformly stretched by the ratio of the row blade counts in the actual machine (61/36). All computations use the axial spacing between blade rows from the original design. The 1:1 blade ratio grid used in the following series of calculations is shown in Fig. 6.30, which includes a detail of the gap region between the blades.

The unsteady lift and drag coefficients are again used to determine when convergence to a periodic solution has been attained. The lift and drag forces are resolved in the circumferential and axial directions respectively, not in a coordinate system aligned with blade stagger angle. The evolution of the unsteady vane and rotor lift and drag coefficients during a complete calculation is shown in Figs. 6.31
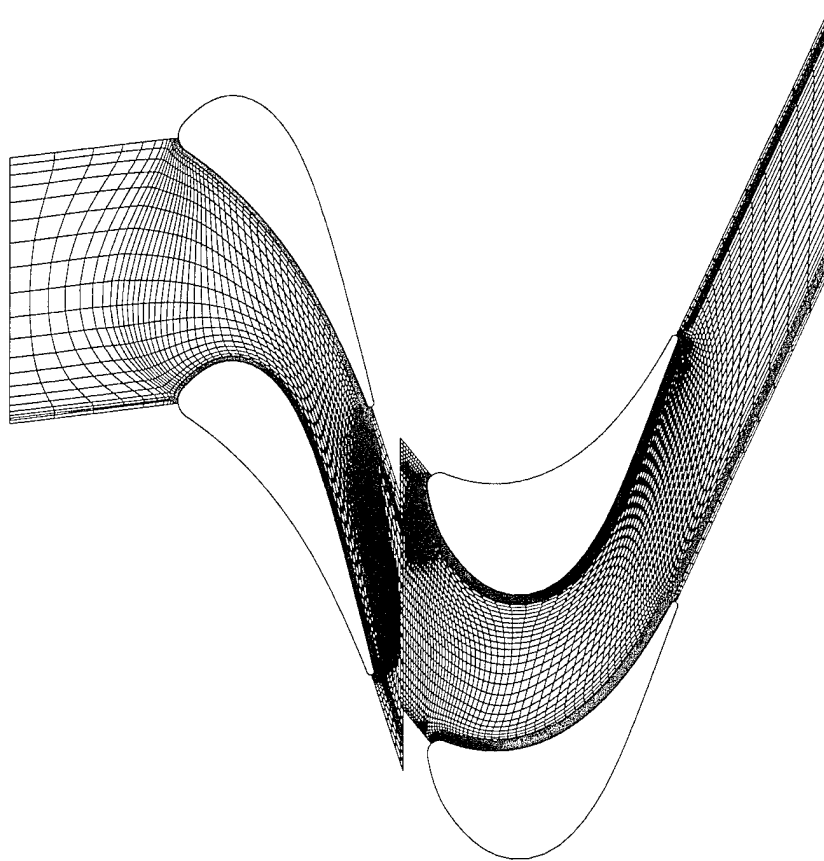
(a) Entire Computational Domain
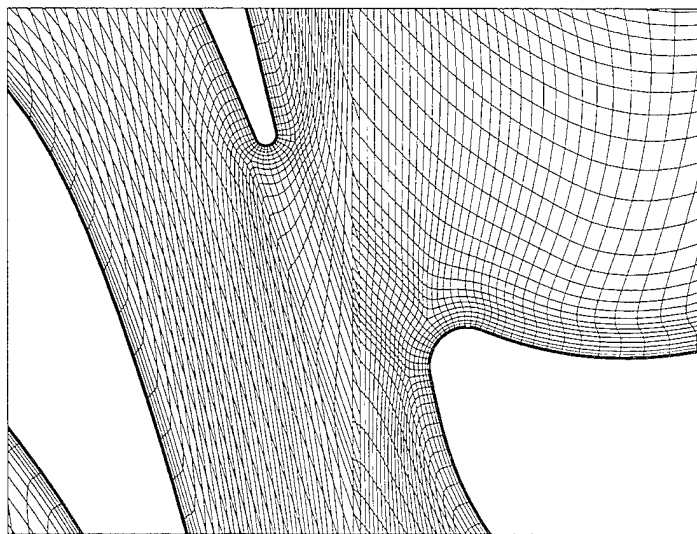
(b) Detail of Gap Region

Figure 6.30: 1:1 blade ratio grid for turbine stage.

for both laminar and turbulent cases, the turbulent solution evolving from a partially converged laminar solution. Both coefficients are defined in the standard form, nondimensionalizing the integrated forces by the inlet vane upstream density and velocity and the respective blade chords. The large coefficient magnitudes are a result of the low inlet velocity ($M_\infty = 0.14$) and the considerable blade forces on a machine with a pressure ratio of 4.3 and more than 70° of turning in each row. This choice of nondimensionalization also leads to the negative lift coefficient on the rotor, which is moving downwards. Figure 6.32 provides a method to reference variations in the lift coefficient to the blade row motion; Mach number contours for the same four blade positions are presented in Fig. 6.33. From these figures the primary effects of the blade motion on the blade forces may be described. At the beginning and end of the blade passing cycle, (points A and E in Figures 6.32 and 6.33), the rotor passage is aligned with the vane passage, causing the least blockage to the flow. Because the stage mass flow is essentially constant throughout the cycle, the low blockage causes the flow velocity along the blade suction surfaces to drop, lowering the lift forces. Similarly, at position C, both the rotor alignment with the center of the vane passage and the vane wake flow combine to maximize the blockage. This increases the flow velocity across the suction sides of both blades, raising the lift forces. While the vane lift force is primarily affected by the passing potential field of the rotor, the vane wake produces an additional effect on the rotor lift. This is evident at position D, where the vane wake fluid forms a large vortex along the rotor pressure surface, further increasing the passage blockage.

A number of issues must be resolved when calculating a realistic geometry, particularly considering the effect on the required computational time. The object of any calculation is to provide the most accurate representation of the physical system in the least computational time. Therefore, it is important to understand the sensitivity of the computed solution to the choice of the various computational
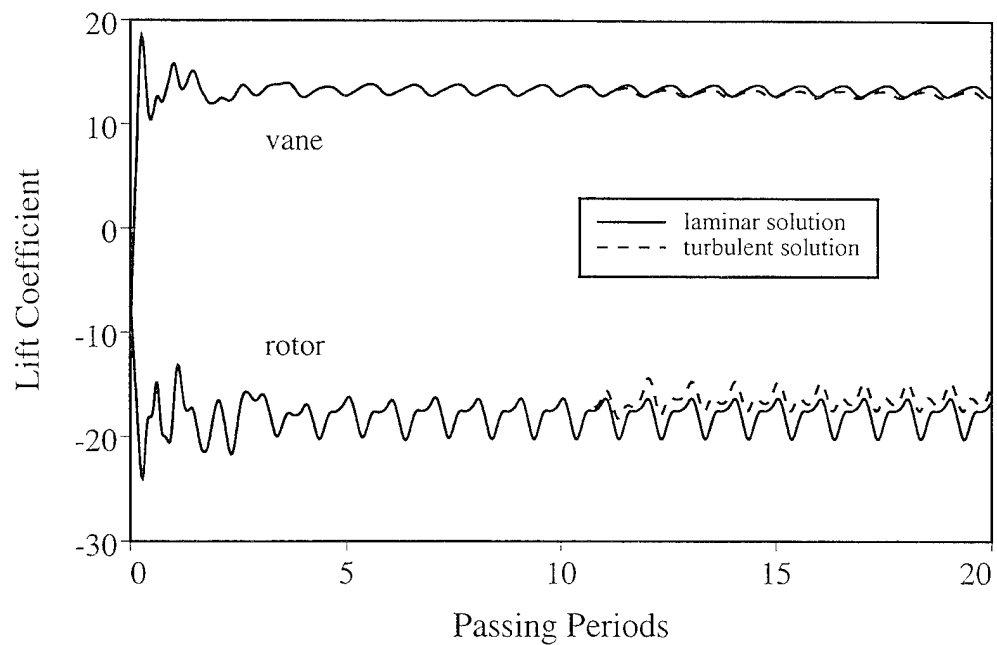
(a) lift coefficient
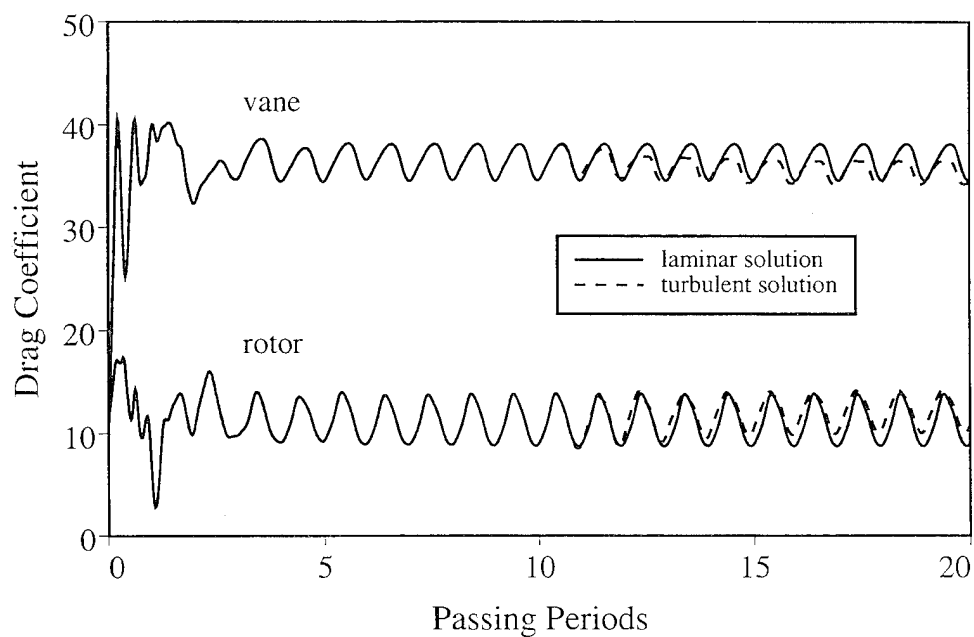
(b) drag coefficient

Figure 6.31: Evolution of lift and drag coefficients for fully turbulent solution (50 iterations per passing, 2 orders-of-magnitude residual drop).

128
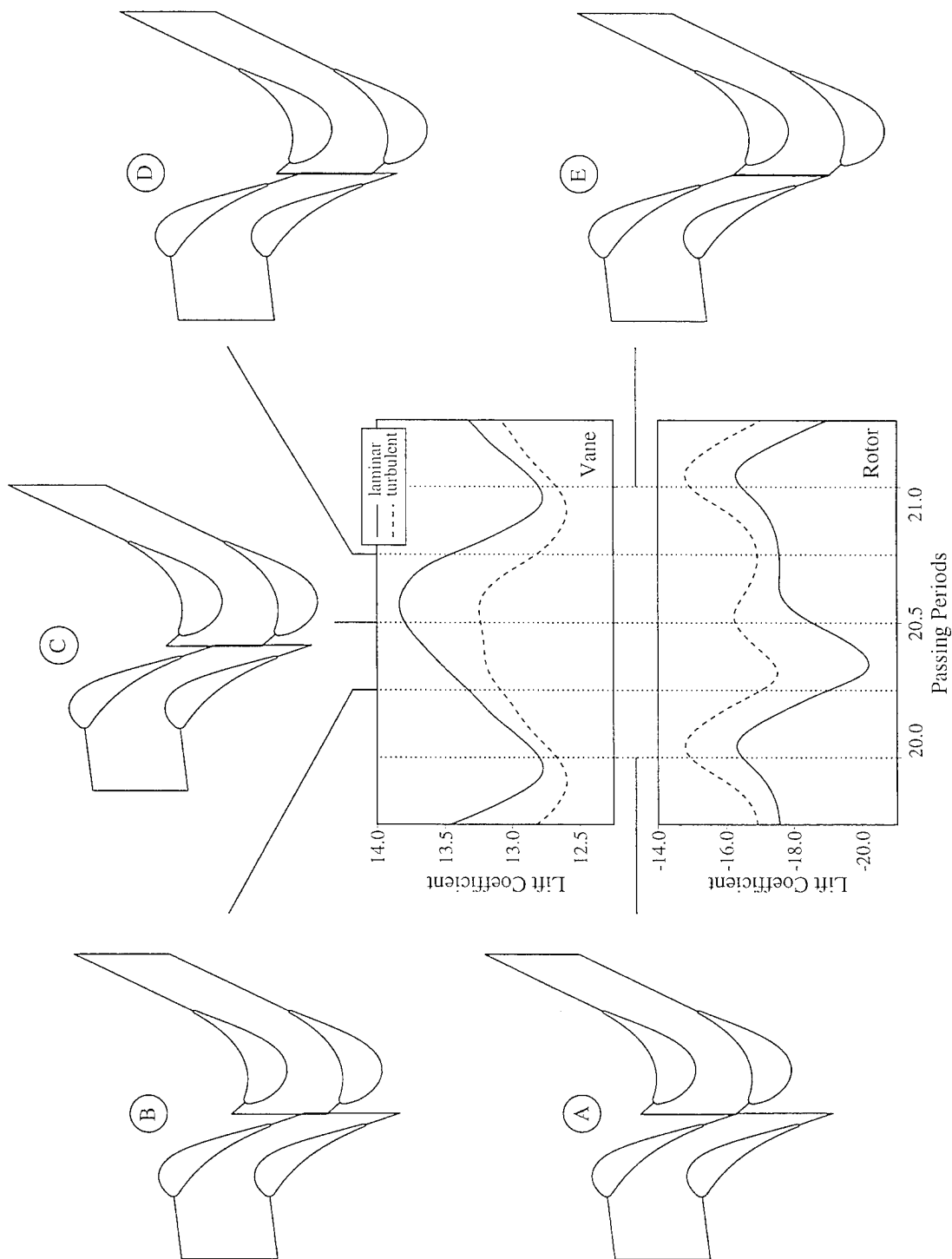
Figure 6.32: Vane and rotor relative positions throughout a single blade passing.
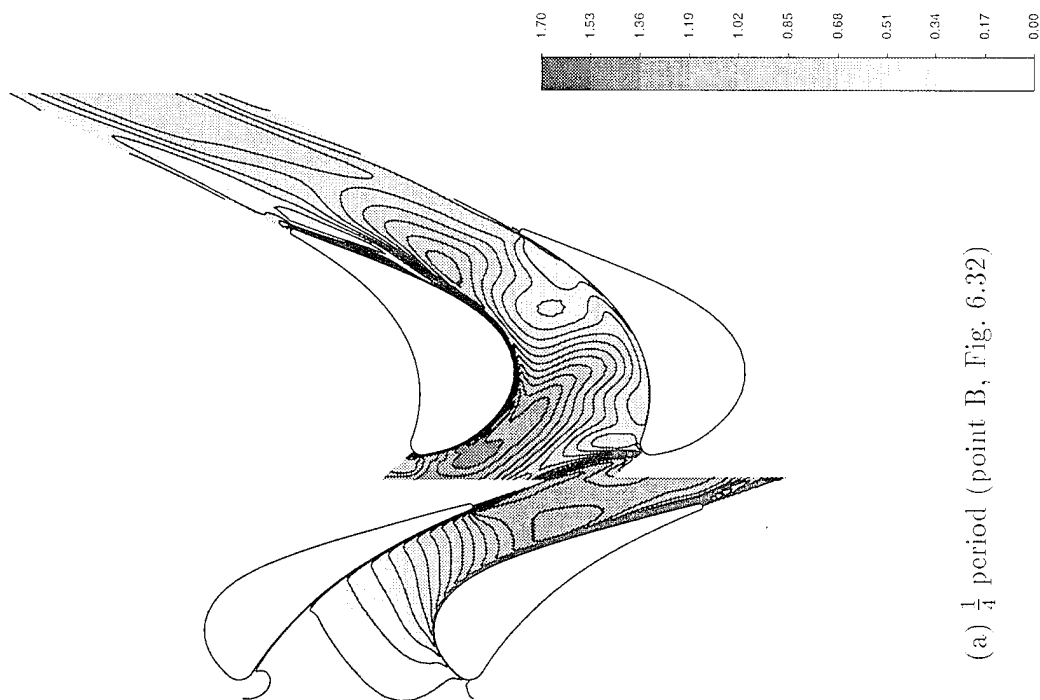
(b) $\frac{1}{2}$ period (point C, Fig. 6.32)

(a) $\frac{1}{4}$ period (point B, Fig. 6.32)

Figure 6.33: See caption page 131.

(c) $\frac{3}{4}$ period (point D, Fig. 6.32)

(d) full period (points A and E, Fig. 6.32)
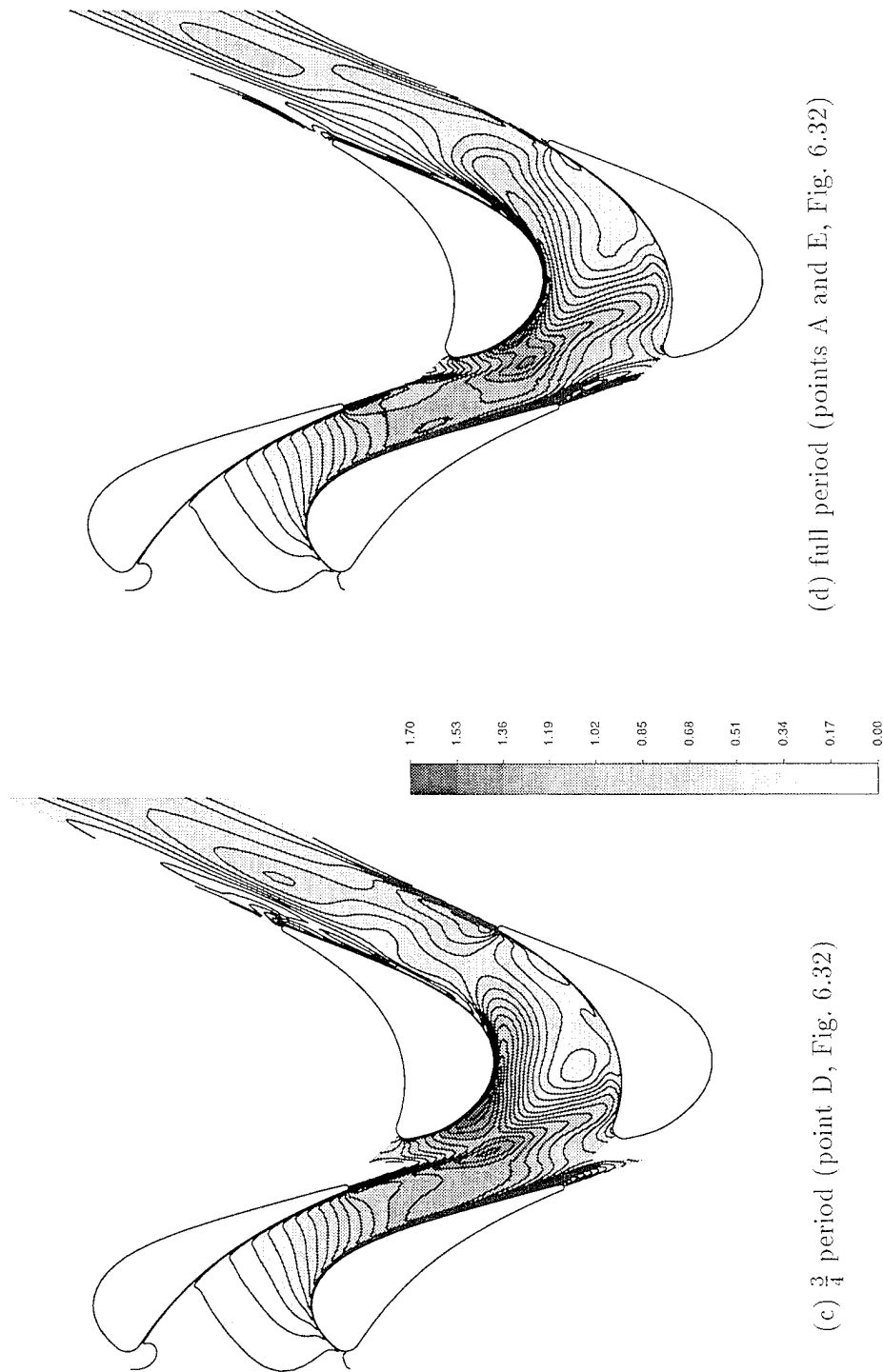
Figure 6.33: Mach number contours for full blade passing period; turbulent solution, 200 iterations per passing.

parameter. As will be shown below, each of these choices has an effect on both the solution and the time required for the computation. These issues will be discussed using calculations with the coarse grid shown in Fig. 6.30, before adapted grid results are presented. The questions to be considered include the following:

- How many real time steps are required per blade passing period?

- How well do the results from the implicit solution compare to a fully explicit solution?

- How many orders-of-magnitude drop in the residual are required at each real time step?

- How does the implementation of the turbulence model affect the solution?

The first set of calculations were performed to examine the effect of the temporal resolution on the computed solution. Coarse grid solutions were obtained for 25, 50, 100, 200, and 400 real time iterations per rotor passing, each solution obtained with two orders-of-magnitude residual drop per iteration. Both laminar and turbulent calculations were performed, with all solutions starting from a nearly converged 50 iteration per passing solution and allowed to run until periodicity was attained. A comparison of the unsteady lift and drag coefficients on the vane and rotor is shown in Figs. 6.34 and 6.35 for the laminar calculation, and in Figs. 6.36 and 6.37 for turbulent flow. As noted in the unsteady vortex shedding and hot streak calculations presented in the previous sections, further reduction in the real time step size beyond 200 iterations per passing has little effect on the solution.

In order to evaluate the accuracy of the implicit solver, fully explicit calculations were also performed. With the density of the grid used in these calculations, the global minimum time step was approximately 850 iterations per blade passing.
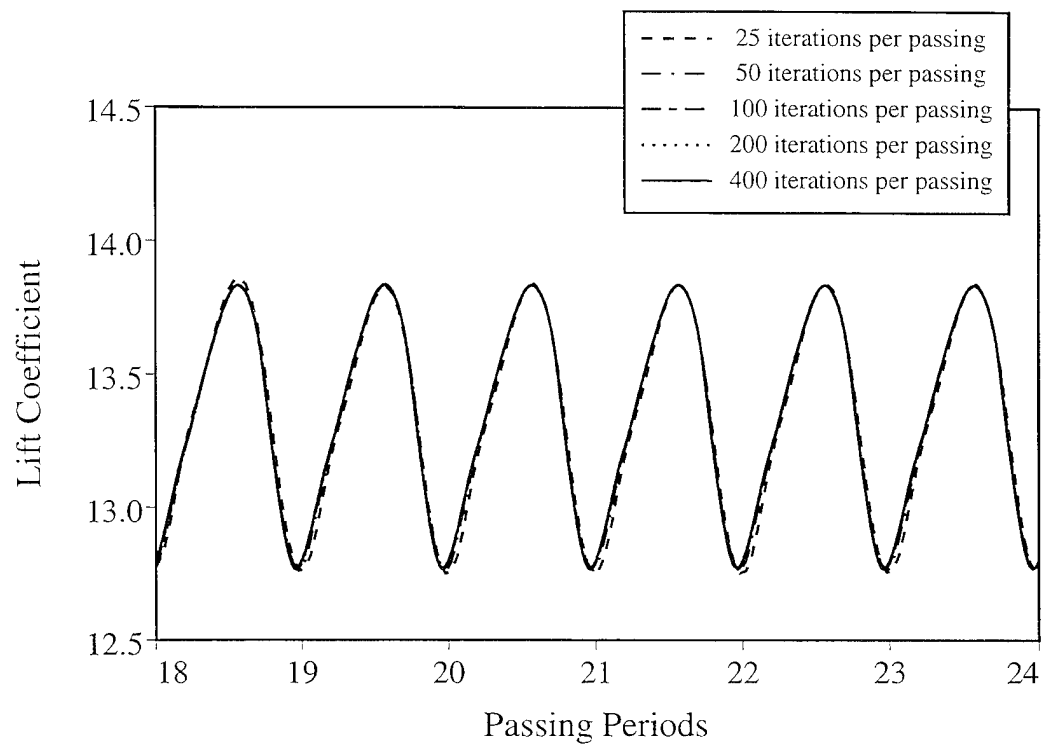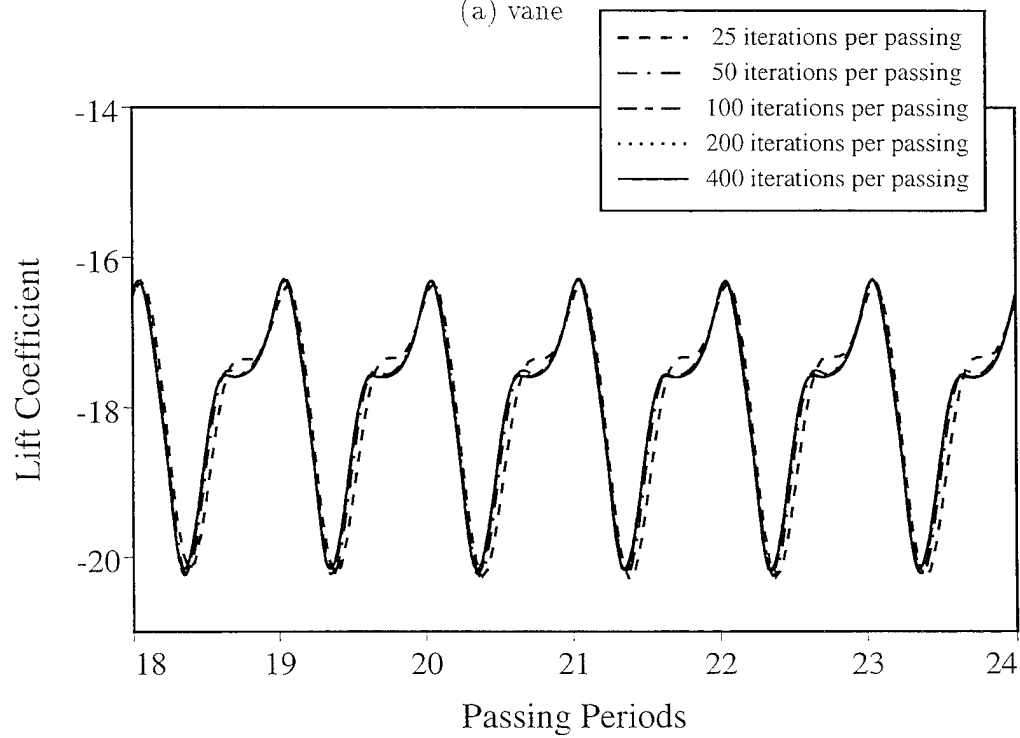
Figure 6.34: Variation of vane and rotor lift coefficient for different iteration counts; laminar solution.
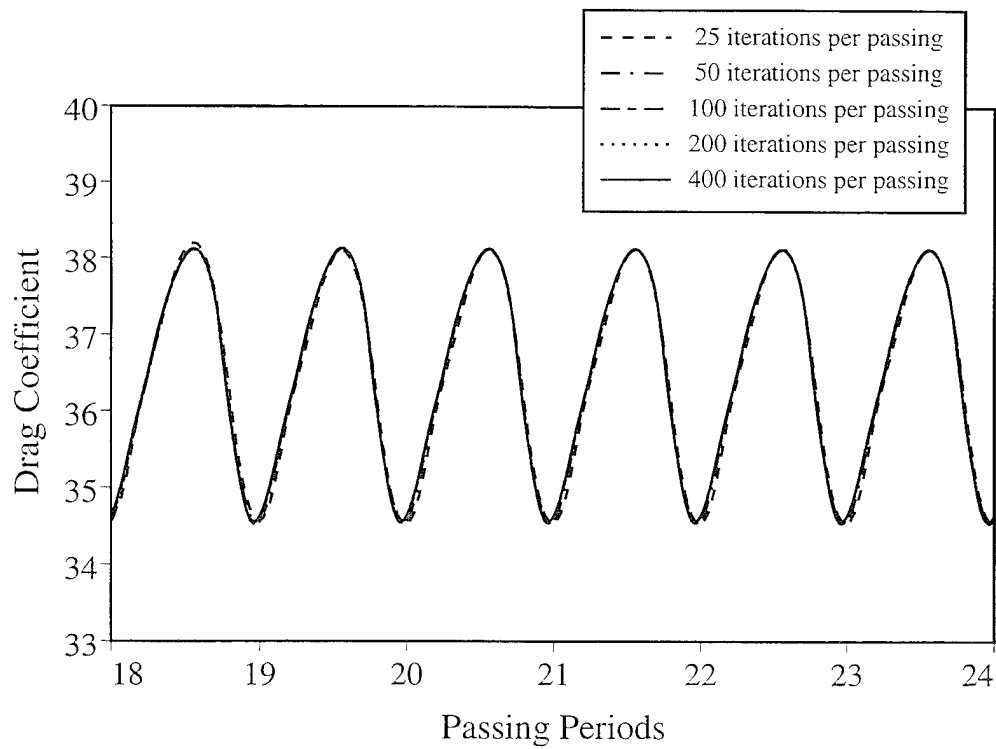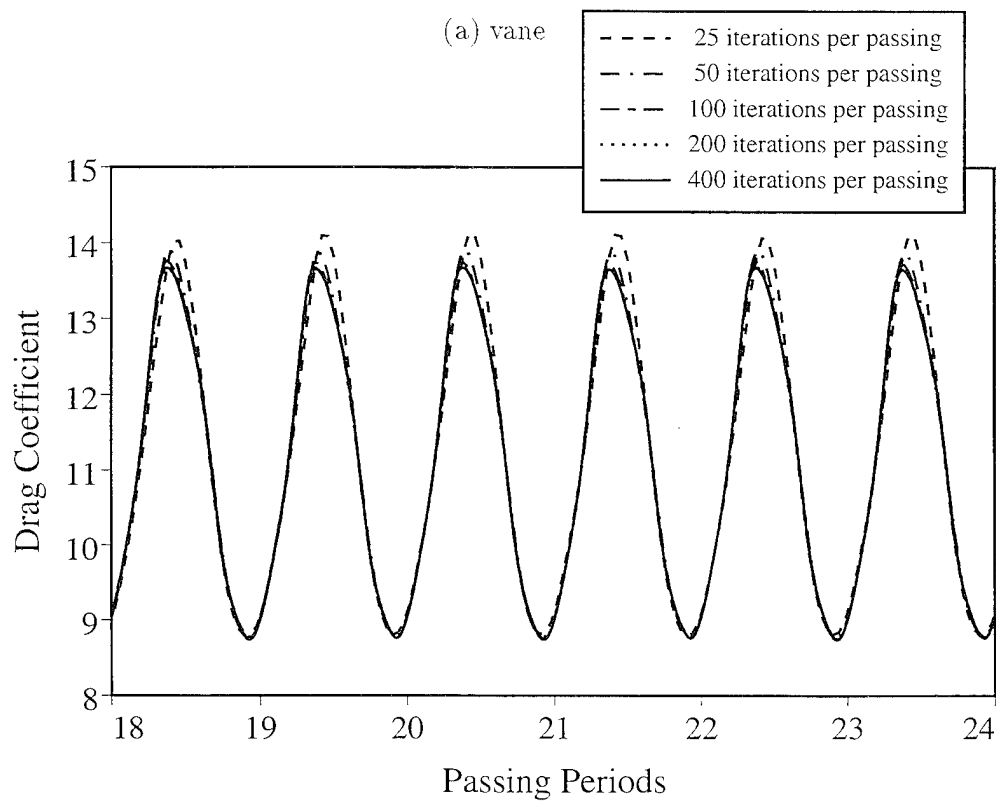
(a) vane

(b) rotor

Figure 6.35: Variation of vane and rotor drag coefficient for different iteration counts; laminar solution.
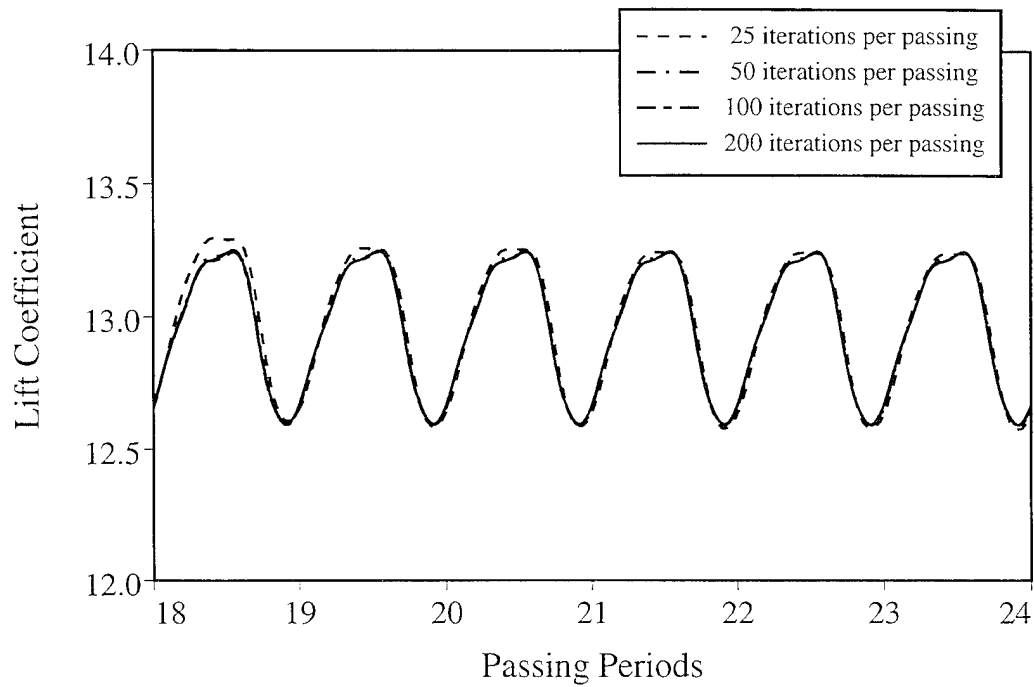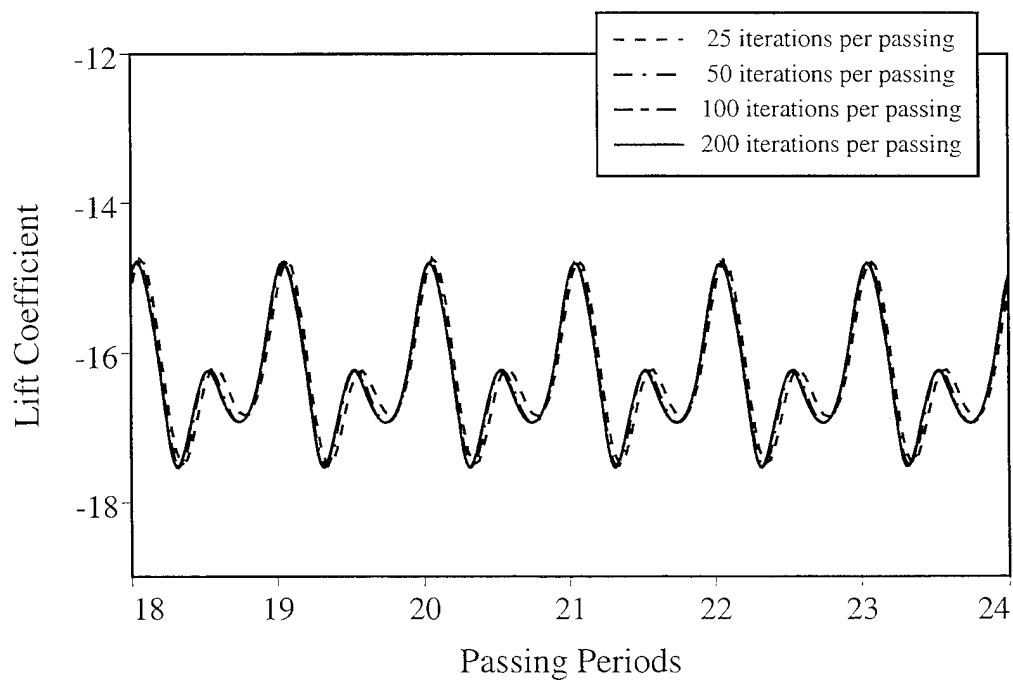
(a) vane

(b) rotor

Figure 6.36: Variation of vane and rotor lift coefficient for different iteration counts; turbulent solution.

135
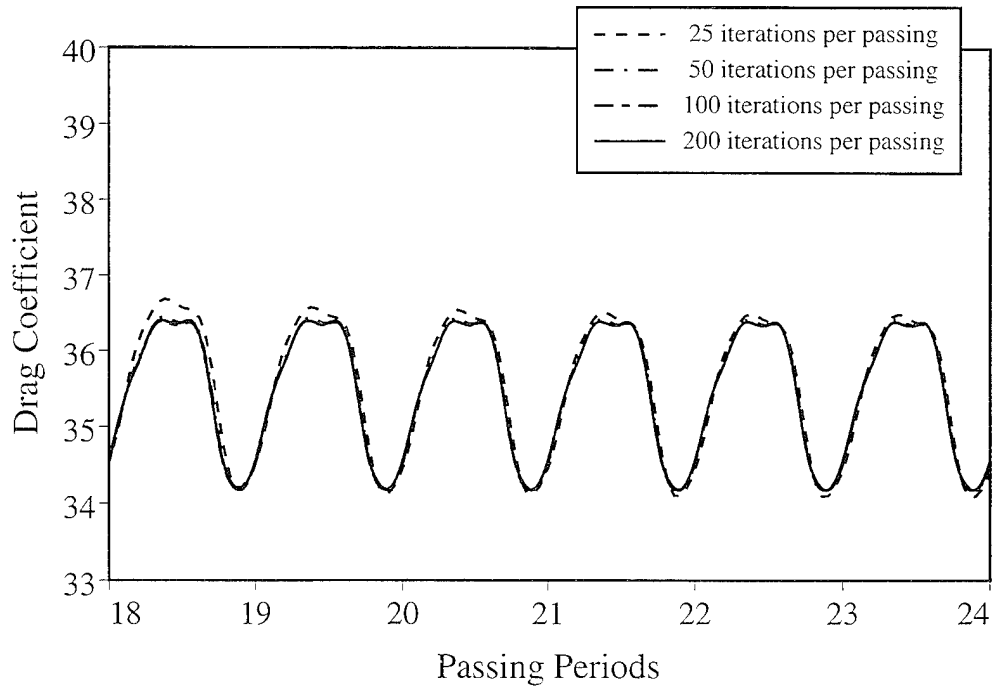
(a) vane
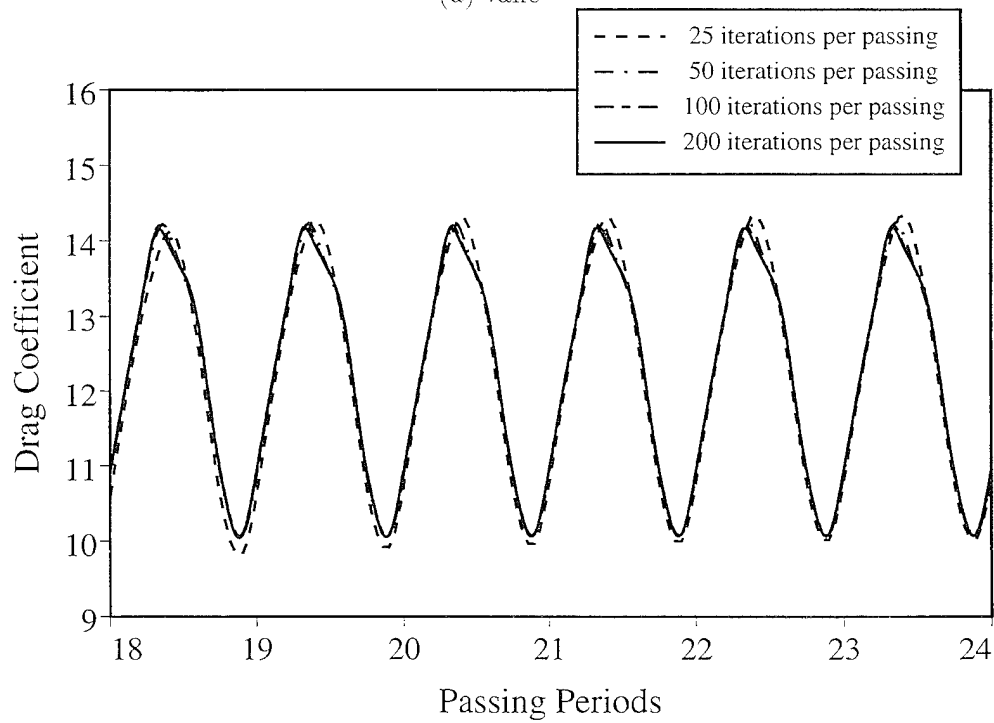
(b) rotor

Figure 6.37: Variation of vane and rotor drag coefficient for different iteration counts: turbulent solution.

136

An additional explicit calculation, with 2500 iterations per passing was also performed to investigate the level of convergence of the explicit solution. A comparison of the explicit solutions with the results of the 50 and 200 iteration per passing cases, for both lift and drag coefficients, is presented in Figs. 6.38 and 6.39. These results indicate a quite acceptable agreement between the implicit and explicit formulations, with little change in the explicit solution as the time step count is increased. A comment about the temporal accuracy of the solution is appropriate here. The modified multi-stage Runge-Kutta algorithm used in the explicit algorithm is of first order temporal accuracy, while the implicit formulation uses a second-order backwards difference temporal operator. Therefore, an explicit solution with 2500 iterations per passing has the same temporal accuracy as the implicit solution with 50 iterations per passing. Comparing solution CPU times for computations *with the same level of temporal accuracy*, there is roughly a factor of 8 savings in computer time between the implicit and explicit solutions.

The statistics of all of these computations, shown in Table 6.2, reflect similar trends as those found during the computation of the cylinder vortex shedding cases. The number of pseudo-time iterations per real time step falls as the real time increment is reduced. It is also interesting to note that the turbulence model has a small effect on the number of pseudo-time iterations required to converge the solution at each real time step, the increased damping introduced by the turbulence model tending to hasten convergence.

The next issue relates directly to the computer time required to obtain a useful solution. If the residual drop used for each real time step does not greatly affect the solution, then considerably fewer pseudo-time iterations, and therefore less computational time, will be required for each blade passing period. In order to evaluate this choice, four solutions were obtained, each with 200 real time iterations per blade passing, at 0.5, 1, 1.5, 2, 3, and 4 orders-of-magnitude residual drop. A comparison
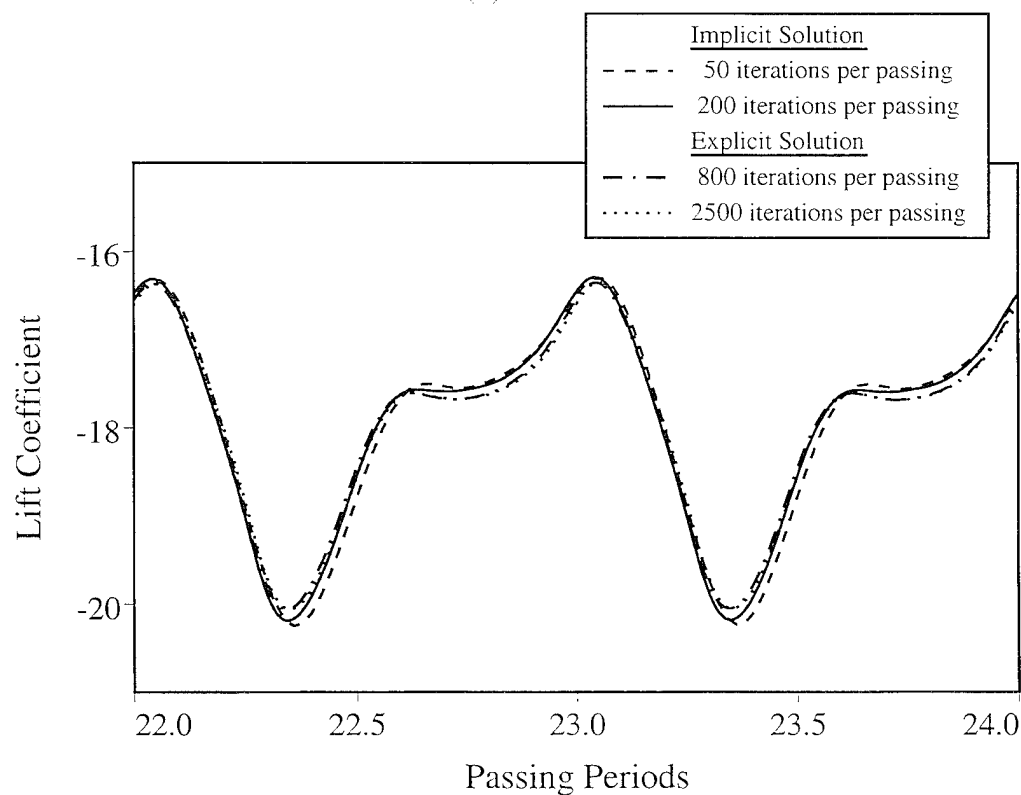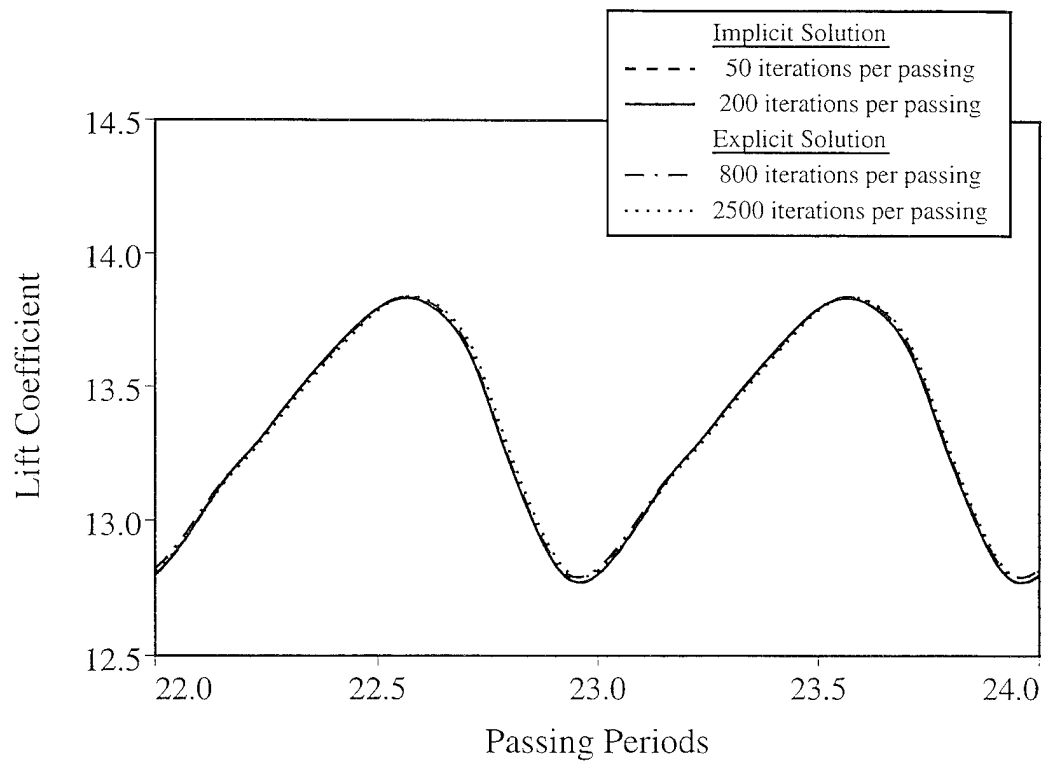
(a) vane

(b) rotor

Figure 6.38: Comparison of lift coefficient for implicit and explicit solutions; laminar solution.
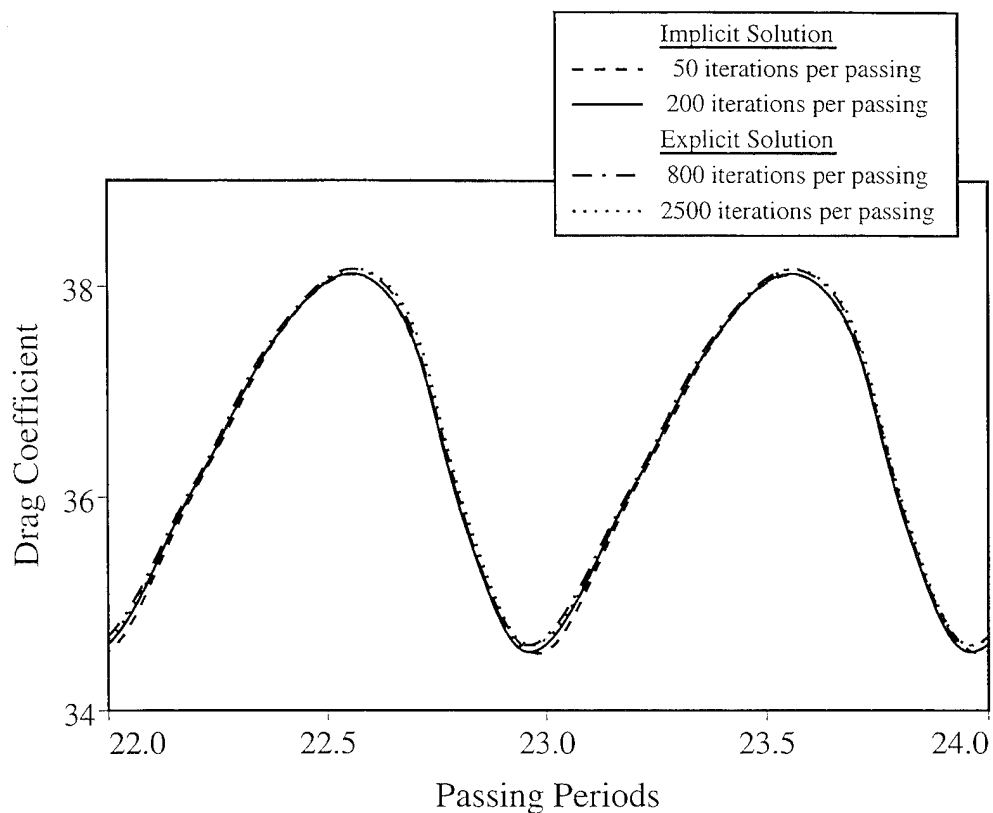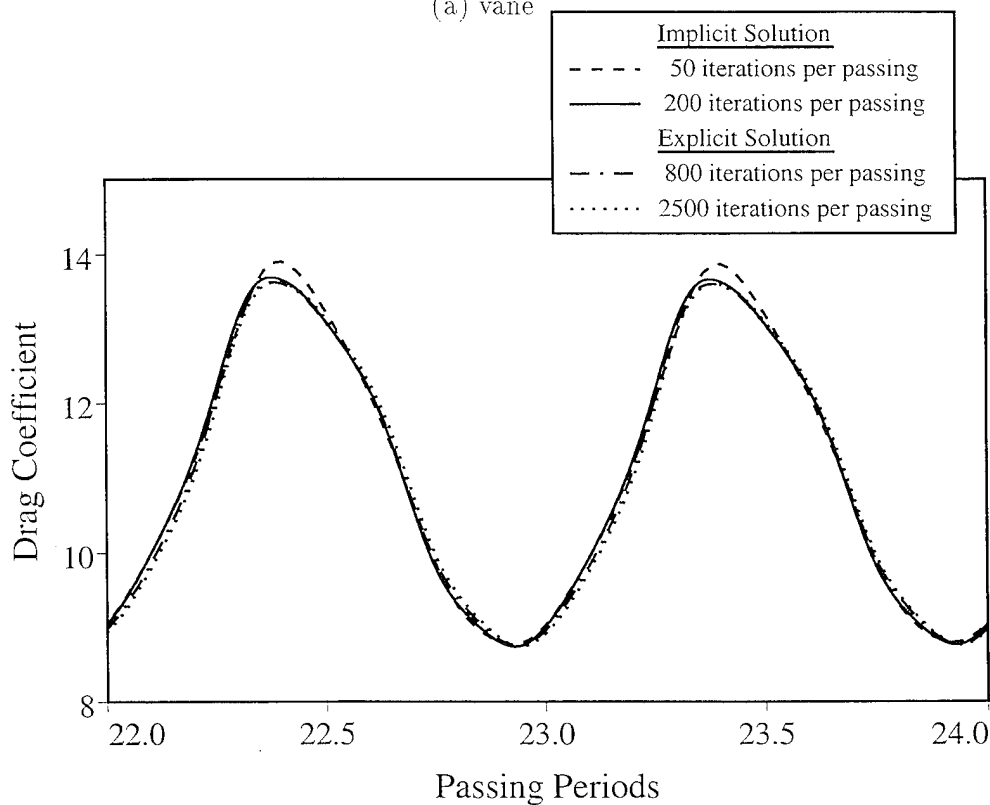
Figure 6.39: Comparison of drag coefficient for implicit and explicit solutions; laminar solution.

139

Laminar Solutions						
real time iterations per period	pseudo-time iterations per period	average iterations per time step	CFL Number			CPU time per period (min)
			max	avg	min	
25	217	8.68	232	27.7	1.86	1.5
50	316	6.33	116	13.8	0.93	2.4
100	415	4.15	58	6.9	0.46	3.2
200	547	2.73	29	3.5	0.23	4.5
400	821	2.05	14	1.7	0.12	6.2
expl	850	1.00		3.5		6.2
expl	2500	1.00		3.5		18.2

Turbulent Solutions						
real time iterations per period	pseudo-time iterations per period	average iterations per time step	CFL Number			CPU time per period (min)
			max	avg	min	
25	240	9.59	259	30.4	1.86	2.3
50	328	6.56	130	15.2	0.92	3.0
100	405	4.05	65	7.6	0.46	3.9
200	513	2.57	32	3.8	0.23	5.6
400	803	2.01	16	1.9	0.12	8.8
expl	850	1.00		3.5		9.3
expl	2500	1.00		3.5		27.4

Table 6.2: Results of iteration count per period tests; 2 orders-of-magnitude residual drop.

of the lift and drag coefficients for the 0.5, 2, and 4 orders-of-magnitude cases are shown in Fig. 6.40 and 6.41. These results clearly show that the choice of residual does not have a great influence on the eventual solution. Additionally, no influence of the residual drop level was observed on the number of blade passing cycles required to achieve a periodic solution; all solutions were essentially periodic after approximately 20 periods. The increased effort required to attain the additional convergence is reflected in the difference in the iteration count per period for the various cases shown in Table 6.3. This table also includes an accounting of the relative error magnitude averaged over a complete period. This error is defined as the difference

in force coefficient values between a solution of some level of convergence and the solution with four orders-of-magnitude residual. These results led to the selection of two orders-of-magnitude residual drop for many of the other calculations in this section.

orders of residual drop	pseudo-time iterations per period	average iterations per time step	average relative error (%)				CPU time per period (min)
			vane		rotor		
			C_L	C_D	C_L	C_D	
0.5	400.3	2.00	0.029	0.041	0.120	0.132	4.79
1.0	400.9	2.01	0.029	0.041	0.120	0.131	4.80
1.5	406.8	2.03	0.028	0.041	0.119	0.130	4.87
2.0	513.0	2.57	0.017	0.024	0.082	0.126	5.64
3.0	765.7	3.83	0.005	0.007	0.024	0.023	7.40
4.0	1249.5	6.25	0.0	0.0	0.0	0.0	11.10

Table 6.3: Results of magnitude of residual drop tests; turbulent solutions.

The final issue to be examined, is the effect of the turbulence model implementation on the solution. For this example, laminar and turbulent solutions are compared, with an additional solution obtained using the one-equation turbulence model, but without propagation of the turbulent eddy viscosity across the interface between blade rows. The no-propagation case uses a standard no-change boundary condition on the eddy viscosity at the exit of the upstream row, with the eddy viscosity reinitialized at the inlet of the downstream row. All calculations are performed with 200 iterations per blade passing and two orders-of-magnitude residual drop. The results of this comparison, given in Figs. 6.42 and 6.43, show that, as expected, the vast majority of the difference between the laminar and turbulent solutions may be attributed to the modeling of the turbulent boundary layers along the individual blades. The propagation of the turbulent eddy viscosity from the
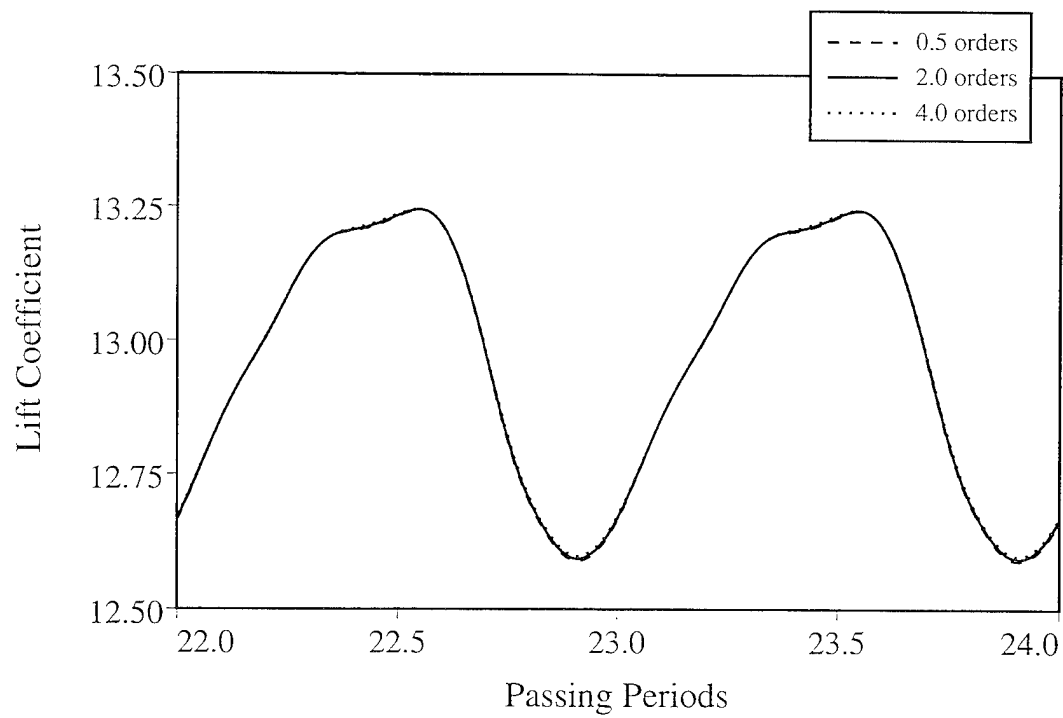
(a) vane
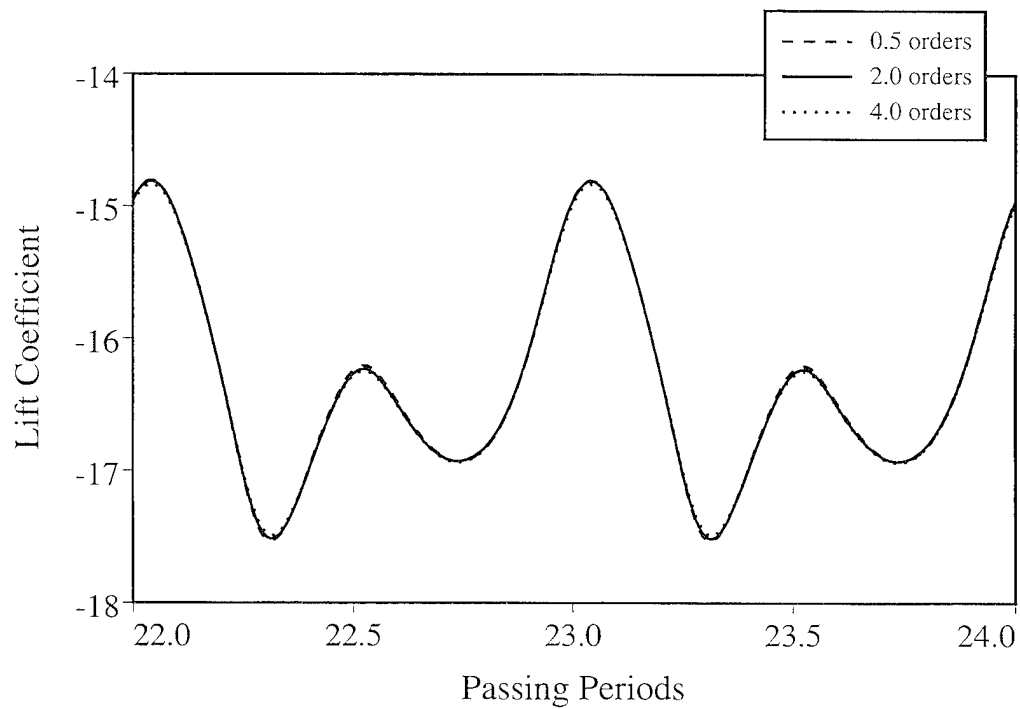
(b) rotor

Figure 6.40: Effect of residual drop on turbine stage lift coefficient (200 iterations per passing).
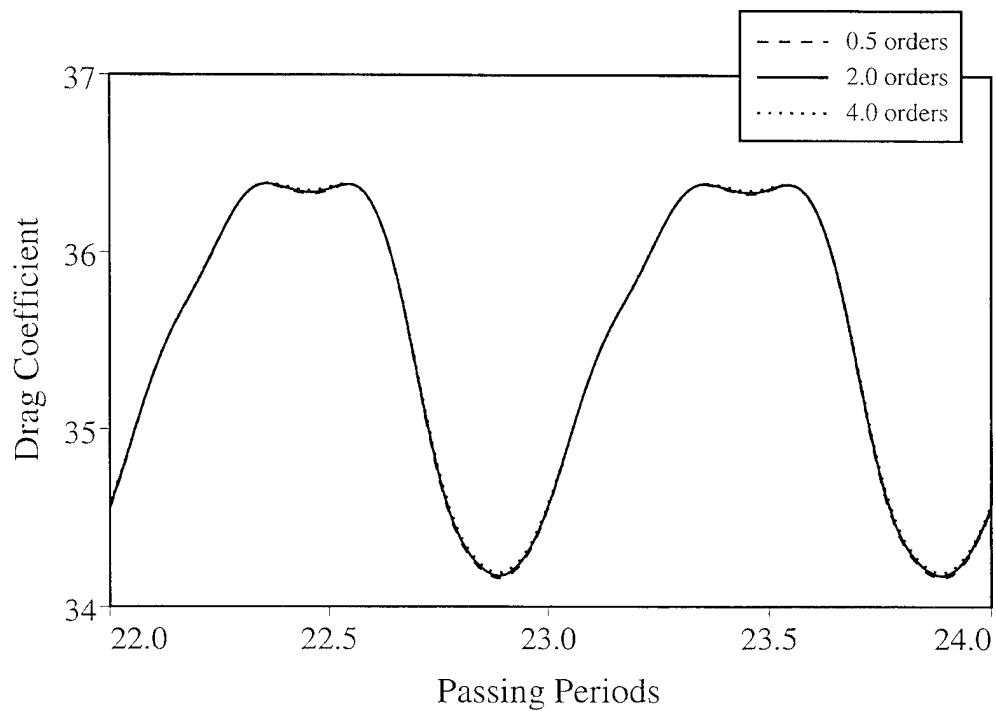
(a) vane
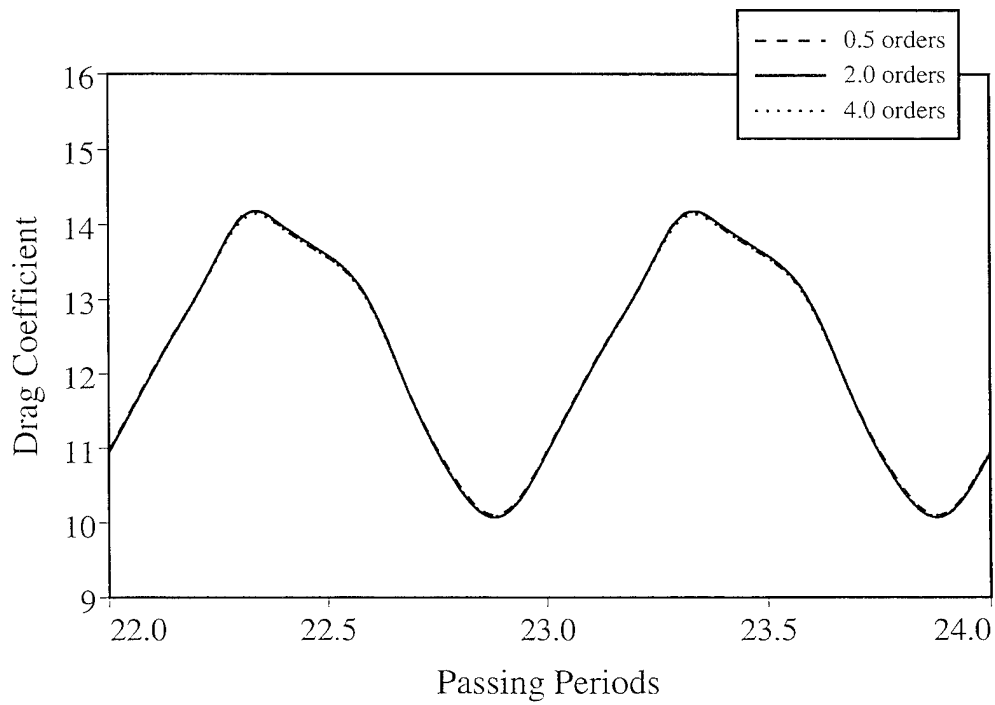
(b) rotor

Figure 6.41: Effect of residual drop on turbine stage drag coefficient (200 iterations per passing).

143

upstream to the downstream row has only a minor effect on the vane, with a more significant effect on the rotor.

The final figures in this section are provided to show the relative level of unsteadiness in the solution and its impact on blade surface parameters. Figures 6.44 and 6.45 show the blade loading and Nusselt number distributions on both blades at ten equally spaced times during a single blade passing period. As expected, the unsteadiness on the vane is confined to the uncovered portion of the blade along the suction surface, upstream of the trailing edge. Little effect is seen on the pressure surface, which is not exposed to the potential field of the passing rotor. The distributions on the rotor clearly show the progression of the vane wake as a wave which progresses along the blade surface. The wake motion is also apparent in the uncovered portion of the rotor suction surface just downstream of the leading edge. These figures highlight the importance of considering not only the time-averaged values, but also the entire envelope of the unsteadiness, particularly for the heat transfer prediction.
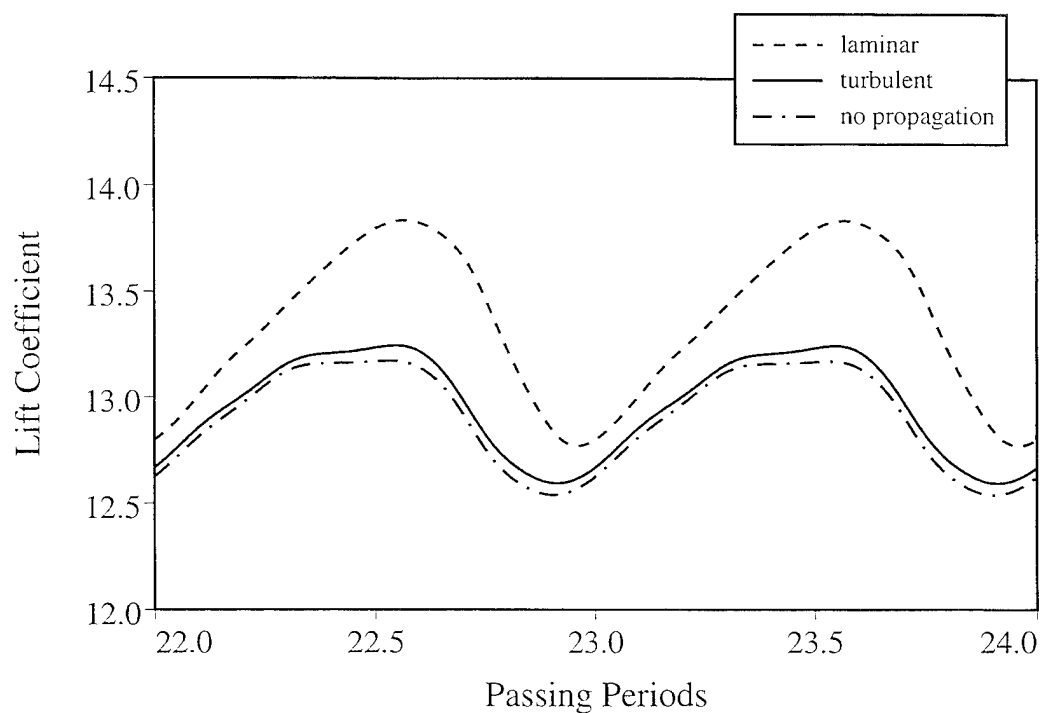
(a) vane
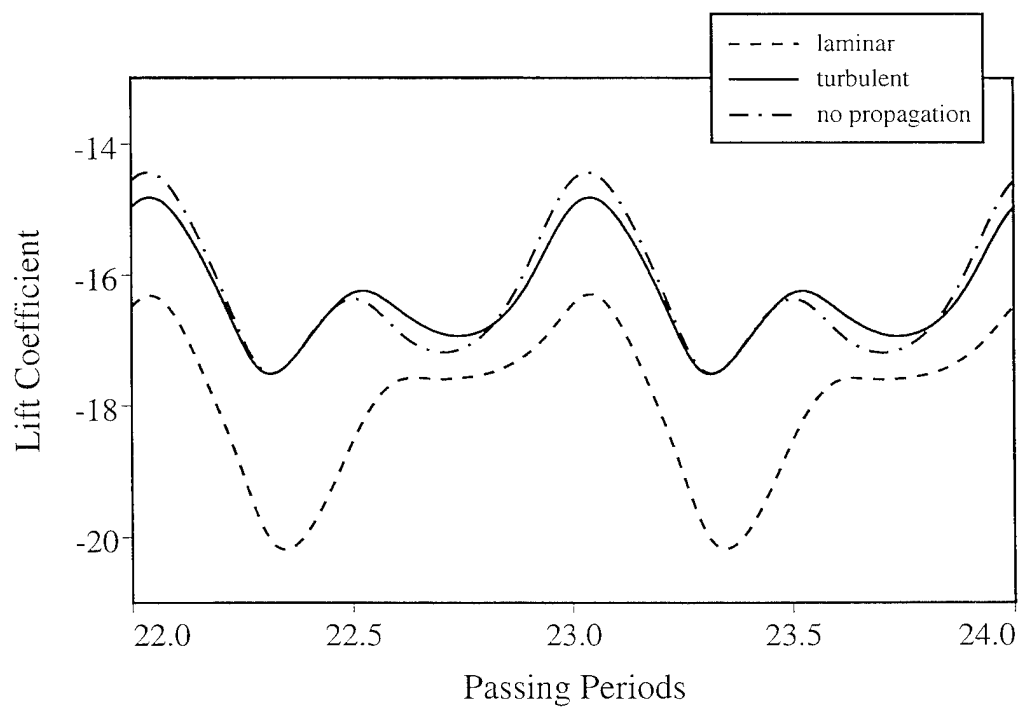
(b) rotor

Figure 6.42: Comparison of lift coefficient for laminar, turbulent, and turbulent without eddy viscosity propagation solutions: 200 iterations per passing.

145
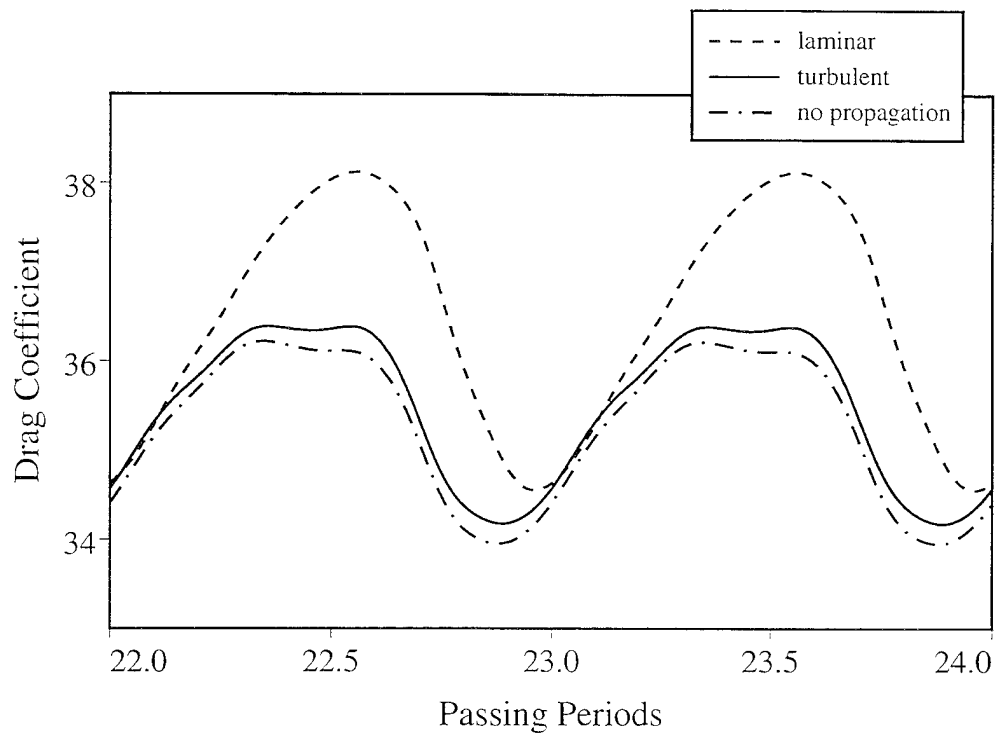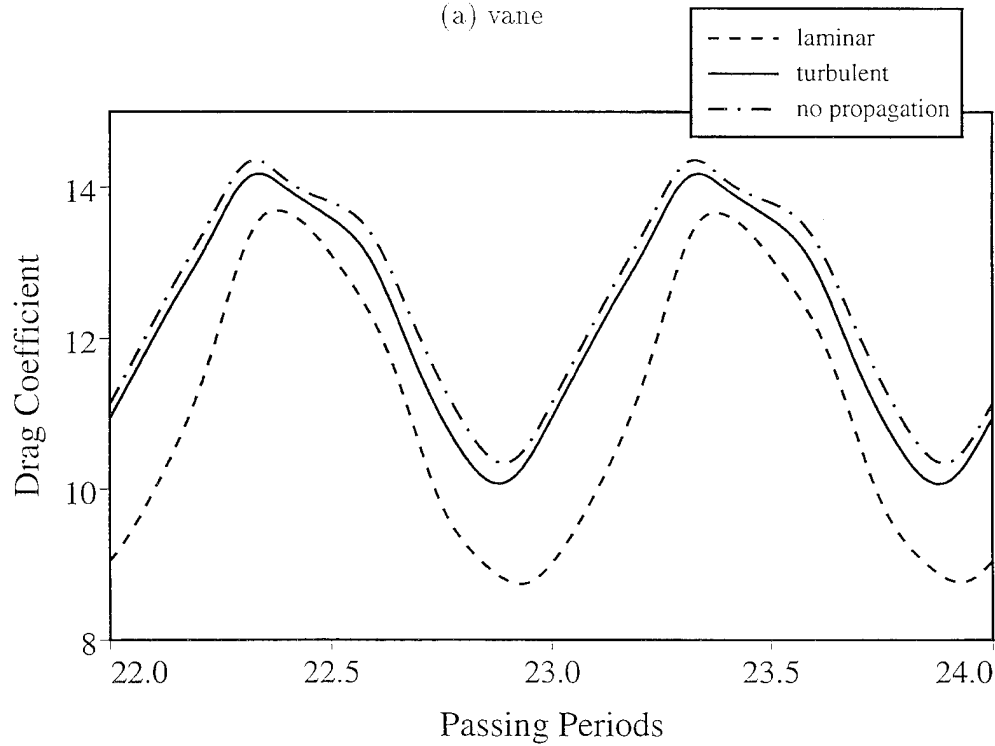
(a) vane

(b) rotor

Figure 6.43: Comparison of drag coefficient for laminar, turbulent, and turbulent without eddy viscosity propagation solutions: 200 iterations per passing.
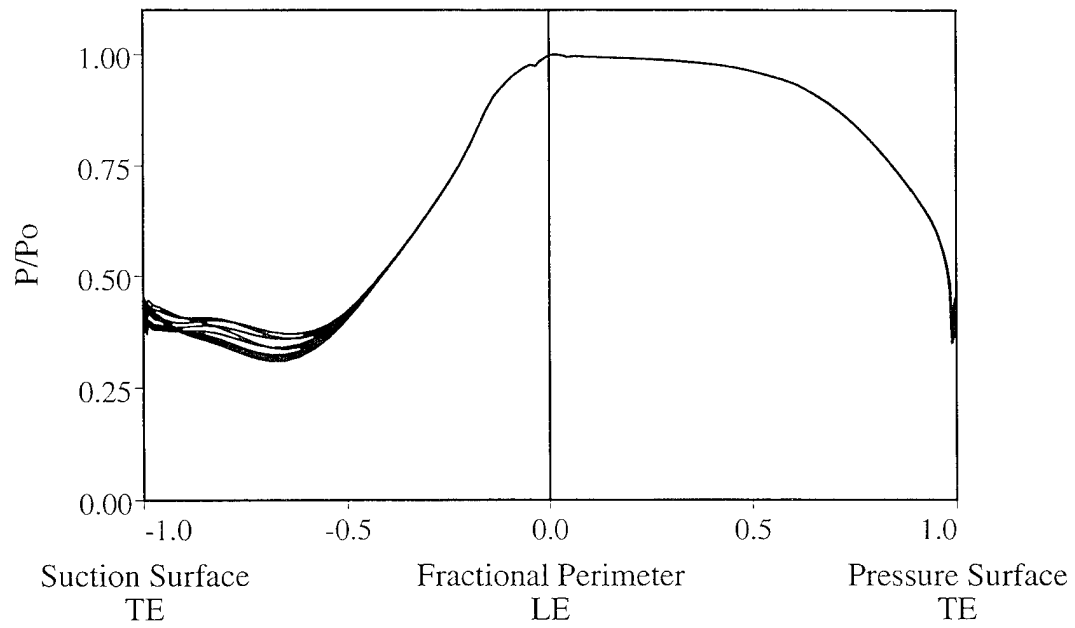
(a) vane
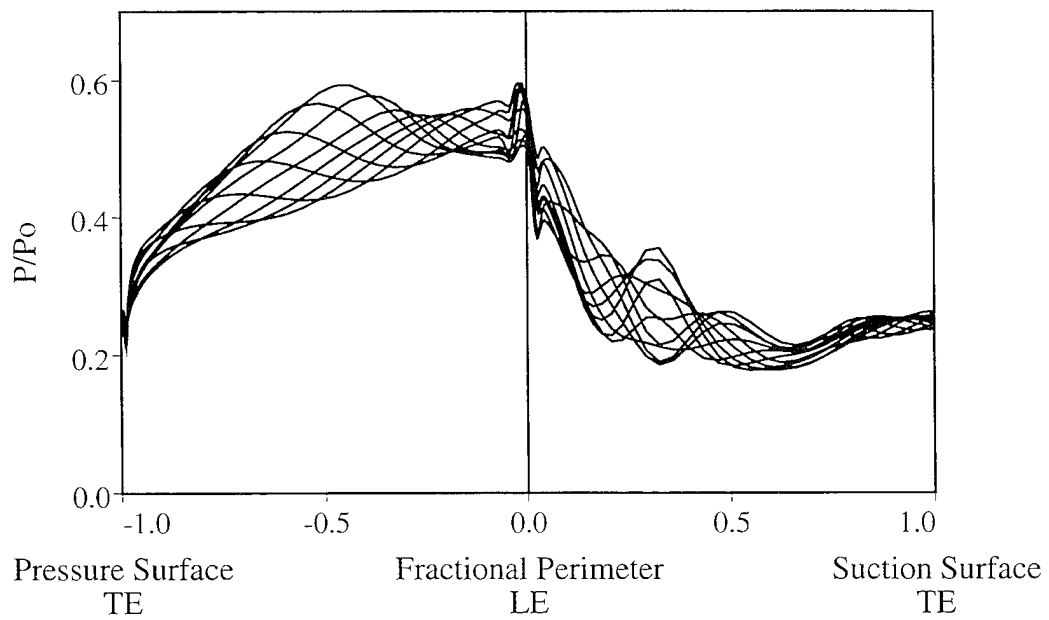
(b) rotor

Figure 6.44: Unsteady blade loading diagrams throughout a rotor passing period.

147

(a) vane
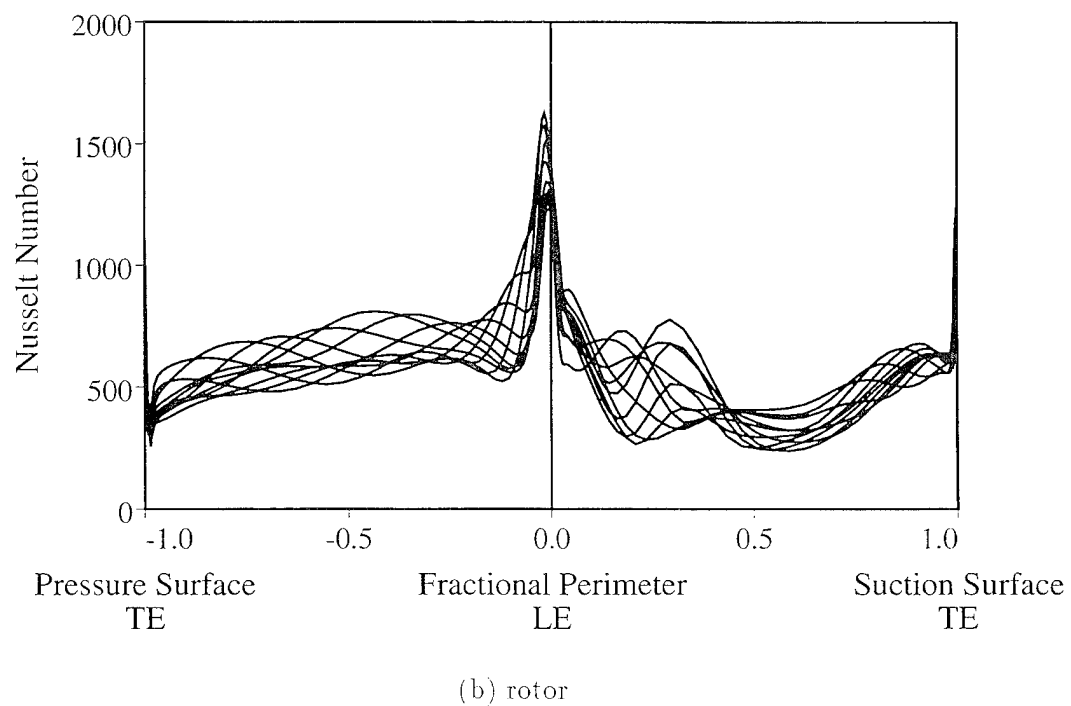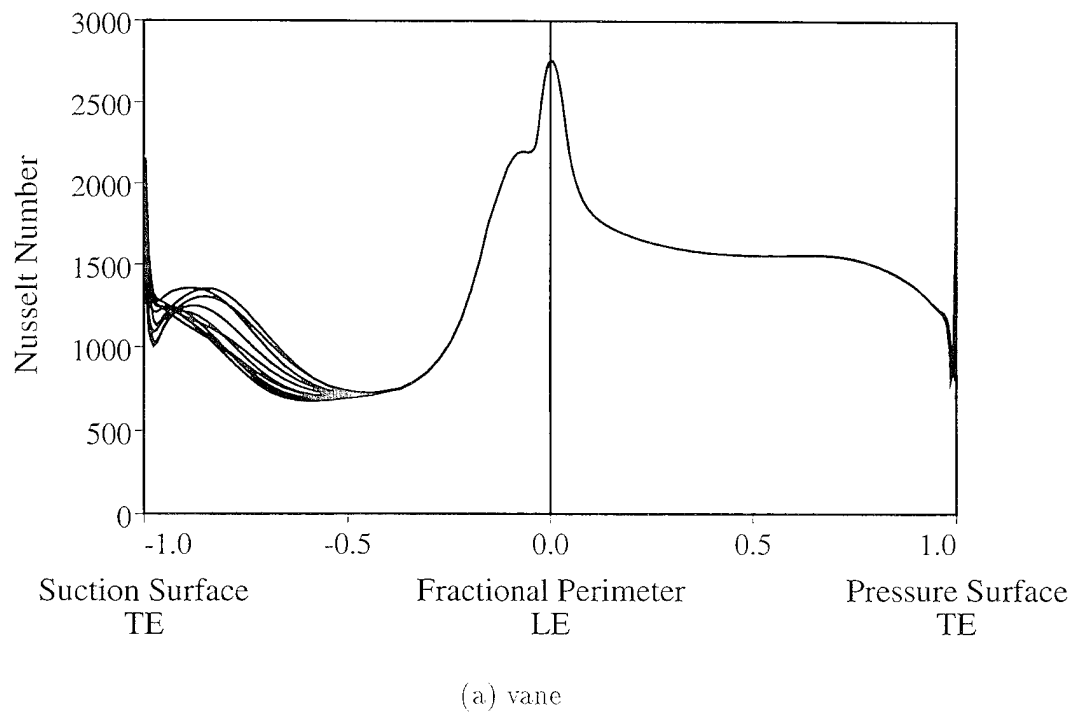
(b) rotor

Figure 6.45: Unsteady blade heat transfer throughout a rotor passing period.

148

6.2.1 Adapted Solution

The final section presents the culmination of this effort, with unsteady solutions of the turbine flow field made using periodic grid adaptation. Starting with the coarse grid described previously, solutions were obtained with a single level of grid adaptation. The 1:1 blade count ratio was again used to reduce the computational cost while serving to highlight the improvement in the solution with the refined grid solution. The adapted solution was started from a previously converged 50 iteration per passing turbulent solution and allowed to re-converge to a periodic solution. As described in the preceding section, a study was performed to determine the number of iterations required for time step independence; all solutions were obtained with two orders-of-magnitude drop in the residuals. Gradients of both turbulent eddy viscosity and relative Mach number were used to trigger grid adaptation, which occurred 20 times per blade passing.

Figures 6.46 - 6.49 show the changes in the computational grid and turbulent eddy viscosity at four times during a blade passing period for the adapted solution. The absolute Mach number and static pressure contours for two of these times are shown in Figs. 6.50 and 6.51. Changes in both the vane and rotor flow fields are quite evident as the adapted portion of the grid tracks the location of the vane wake. As the rotor passes, the accelerating flow around the suction surface thins, and eventually cuts, the vane wake. The downstream portion of the wake then forms a vortex which is deposited on the rotor pressure surface and remains coherent as it is convected through the rotor. It is clear that the adaptation routine has located both the region of rapid acceleration along the uncovered portion of the vane, as well as the pressure surface vortex.

Figures 6.52 and 6.53 show the impact of the grid refinement on the number of iterations required to obtain a time step independent solution. While the coarse

(a) adapted grid

- 0.600E-03
- 0.540E-03
- 0.480E-03
- 0.420E-03
- 0.360E-03
- 0.300E-03
- 0.240E-03
- 0.180E-03
- 0.120E-03
- 0.600E-04
- 0.000E+00
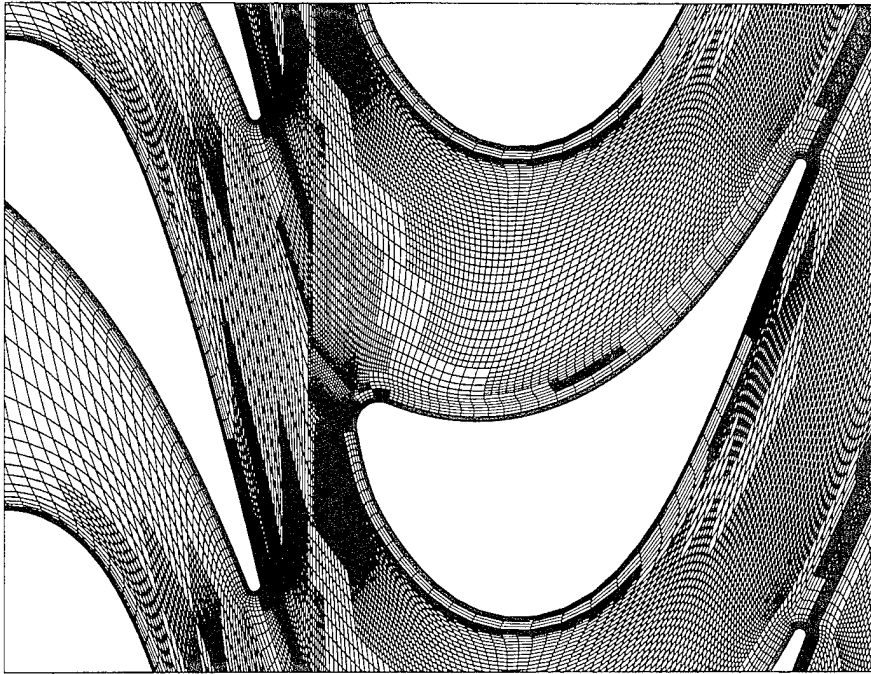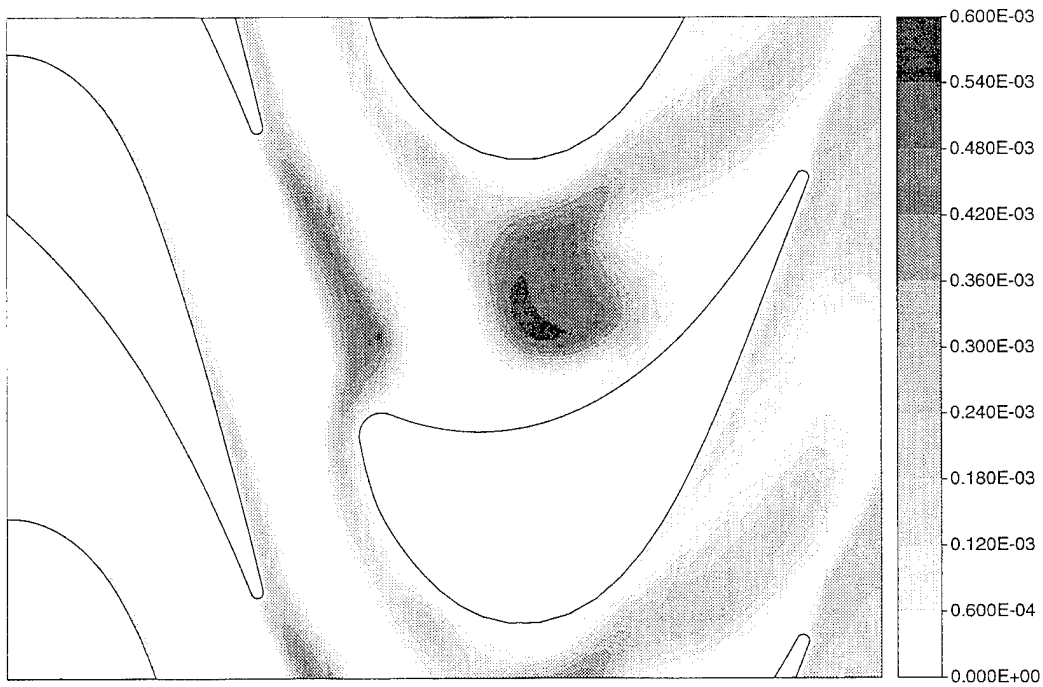
(b) turbulent eddy viscosity

Figure 6.46: Grid detail and turbulent eddy viscosity contours of adapted solution after $\frac{1}{4}$ period; 800 iterations per passing.
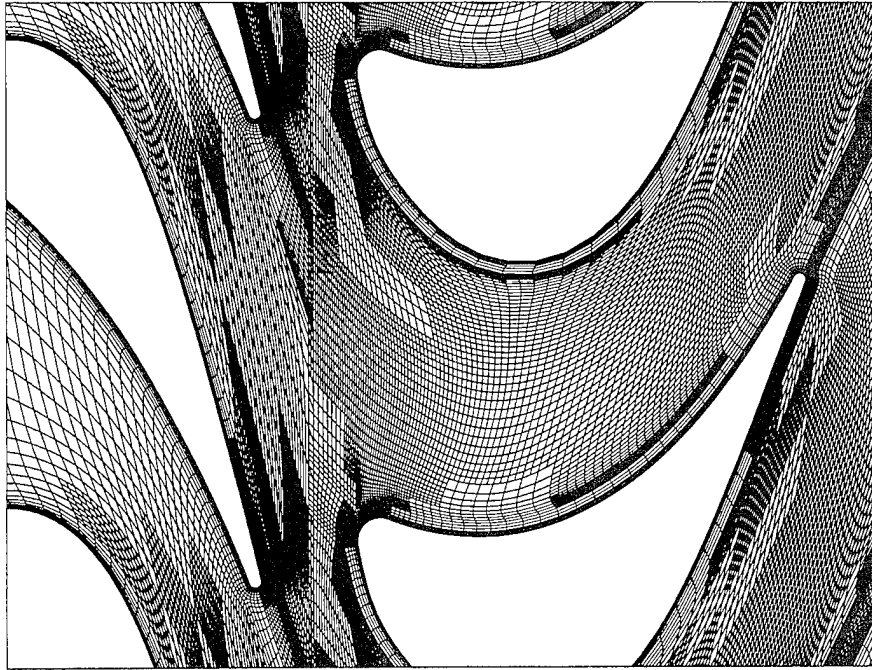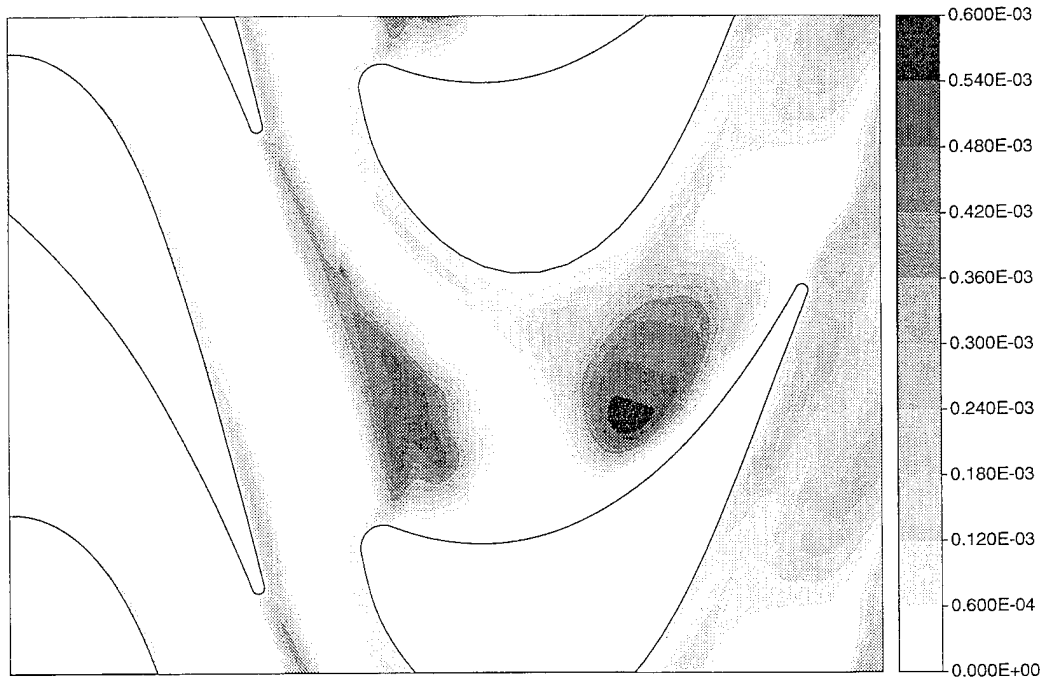
(a) adapted grid

(b) turbulent eddy viscosity

Figure 6.47: Grid detail and turbulent eddy viscosity contours of adapted solution after $\frac{1}{2}$ period; 800 iterations per passing.
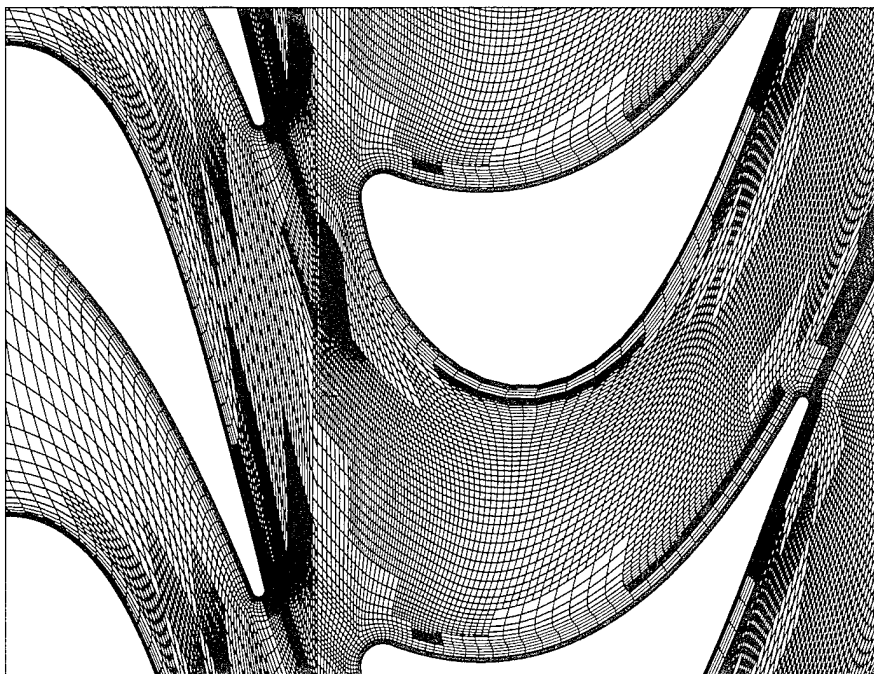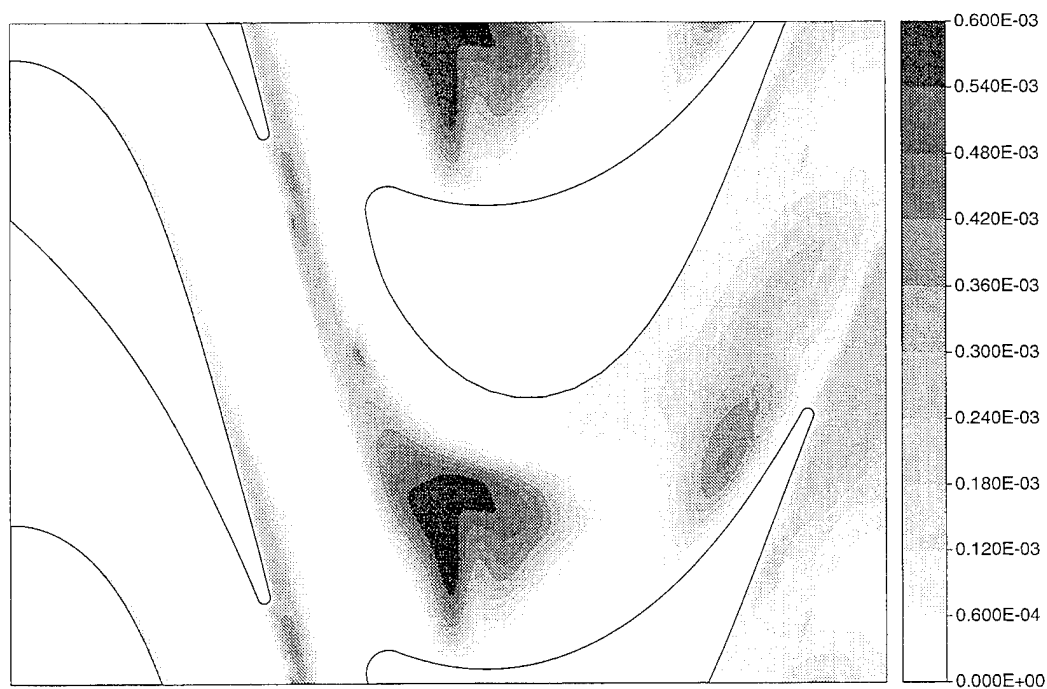
(a) adapted grid

(b) turbulent eddy viscosity

Figure 6.48: Grid detail and turbulent eddy viscosity contours of adapted solution after $\frac{3}{4}$ period; 800 iterations per passing.

(a) adapted grid

0.600E-03
0.540E-03
0.480E-03
0.420E-03
0.360E-03
0.300E-03
0.240E-03
0.180E-03
0.120E-03
0.600E-04
0.000E+00
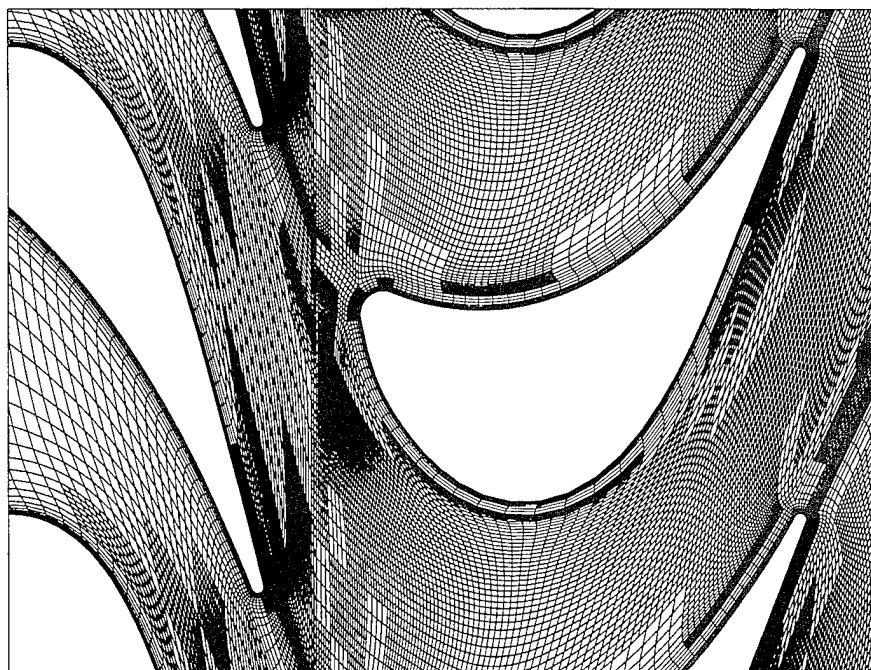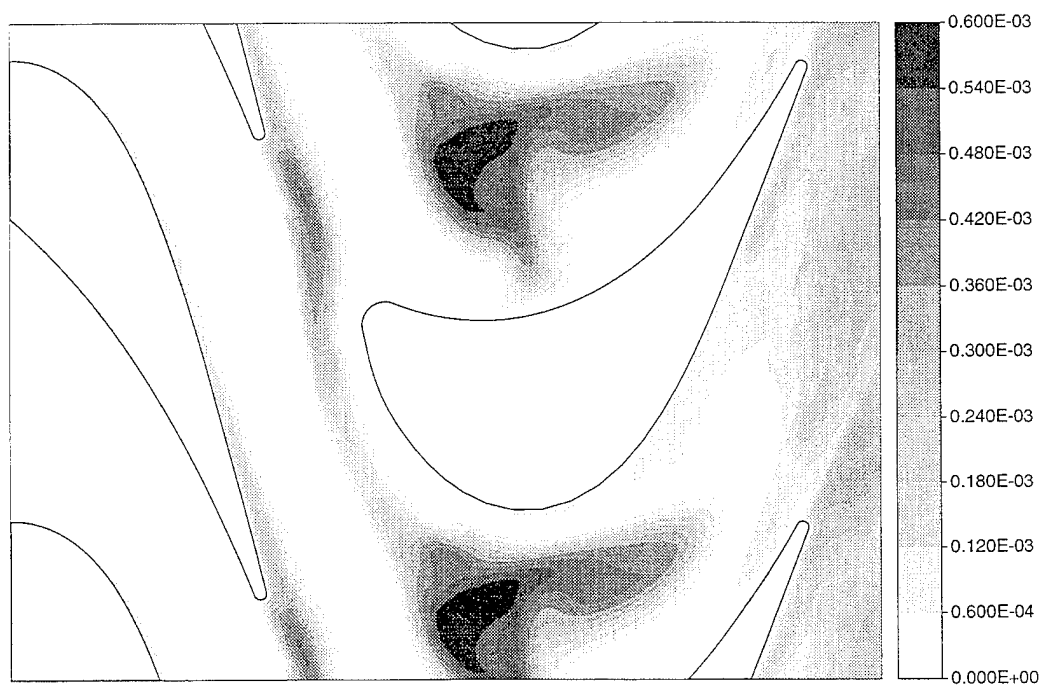
(b) turbulent eddy viscosity

Figure 6.49: Grid detail and turbulent eddy viscosity contours of adapted solution after 1 period; 800 iterations per passing.
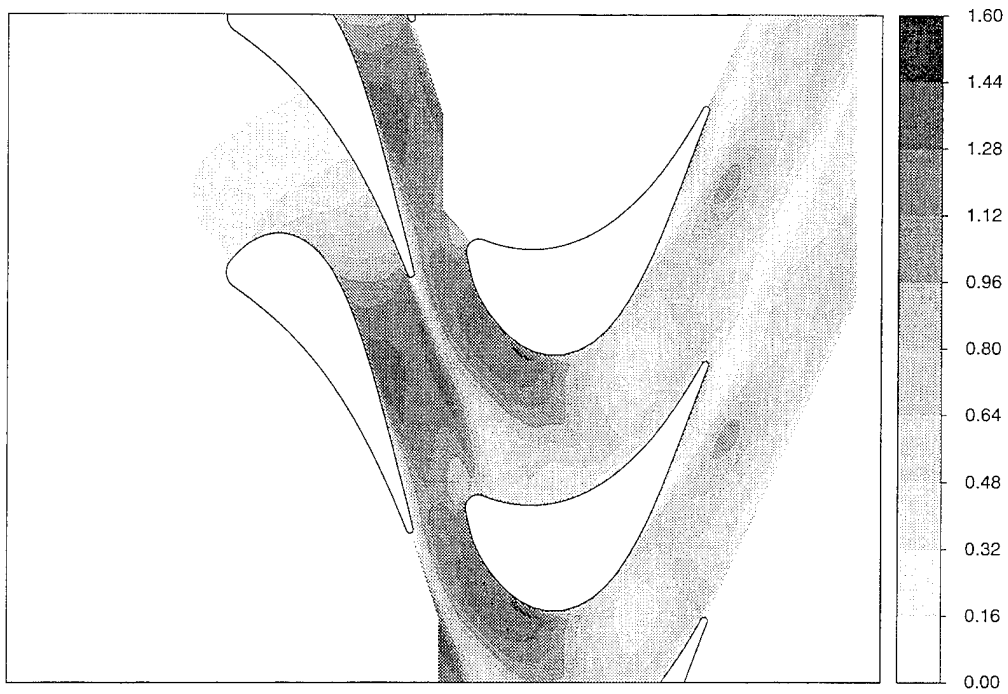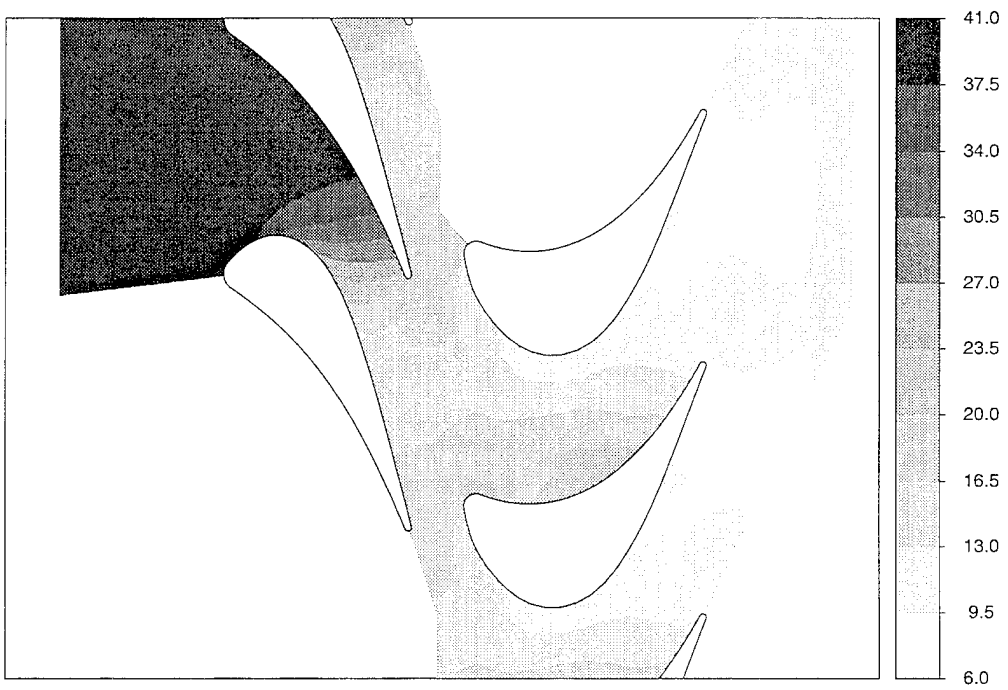
(a) Mach number

(b) static pressure

Figure 6.50: Mach number and pressure contours for adapted solution after $\frac{1}{4}$ period; 800 iterations per passing.
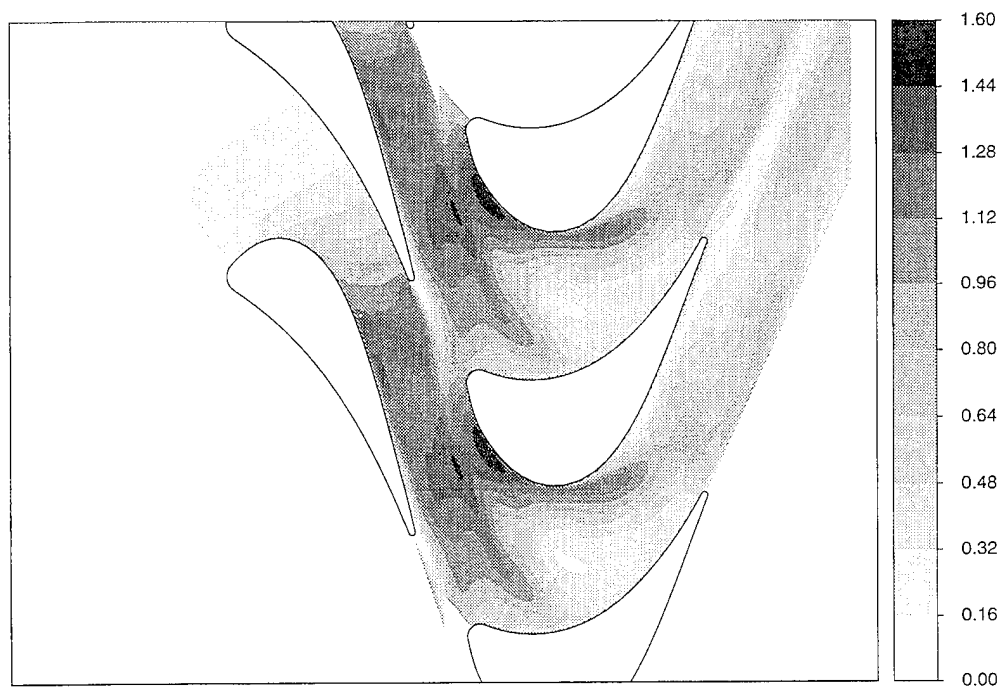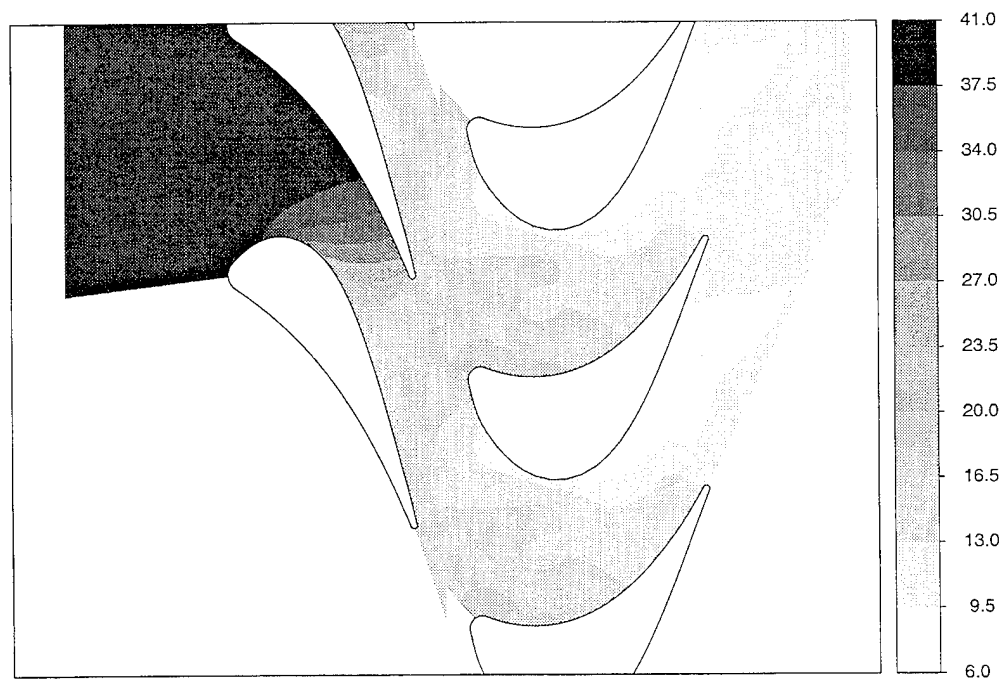
(a) Mach number

(b) static pressure

Figure 6.51: Mach number and pressure contours for adapted solution after $\frac{3}{4}$ period; 800 iterations per passing.

155

grid solution reached an asymptotic solution with only 200 iterations per passing, the adapted solution requires 800 iterations per passing. This result is expected after the experience gained through computing the circular cylinder shedding with the finer grid in an earlier section.

Comparing run times between the coarse grid and adapted solution, there is an approximately four-fold increase in run-time for a single level of grid adaptation. Coarse grid solutions require approximately 6 minutes per period for the 200 iteration per period case, and nearly 80 minutes per period for the adapted solution with 800 iterations per period. The number of points in the solution went from approximately 5500 in the coarse grid, to 17,000 in the adapted solution. As a point of comparison, a uniformly fine grid would contain nearly 26,000 nodes and require an estimated 116 minutes per period for the 800 iteration per passing solution. A summary of the adapted grid solution results is presented in Table 6.4.

Adapted Solutions						
real time iterations per period	pseudo-time iterations per period	average iterations per time step	CFL Number			CPU time per period (min)
			max	avg	min	
100	810	8.1	176	16.2	0.46	30.2
200	1050	5.3	88	7.1	0.23	29.6
400	1610	4.0	44	3.7	0.12	47.0
800	2612	3.3	22	1.9	0.06	76.4

Table 6.4: Results of iteration count per period tests; adapted grid, 2 orders-of-magnitude residual drop.

(a) vane

(b) rotor
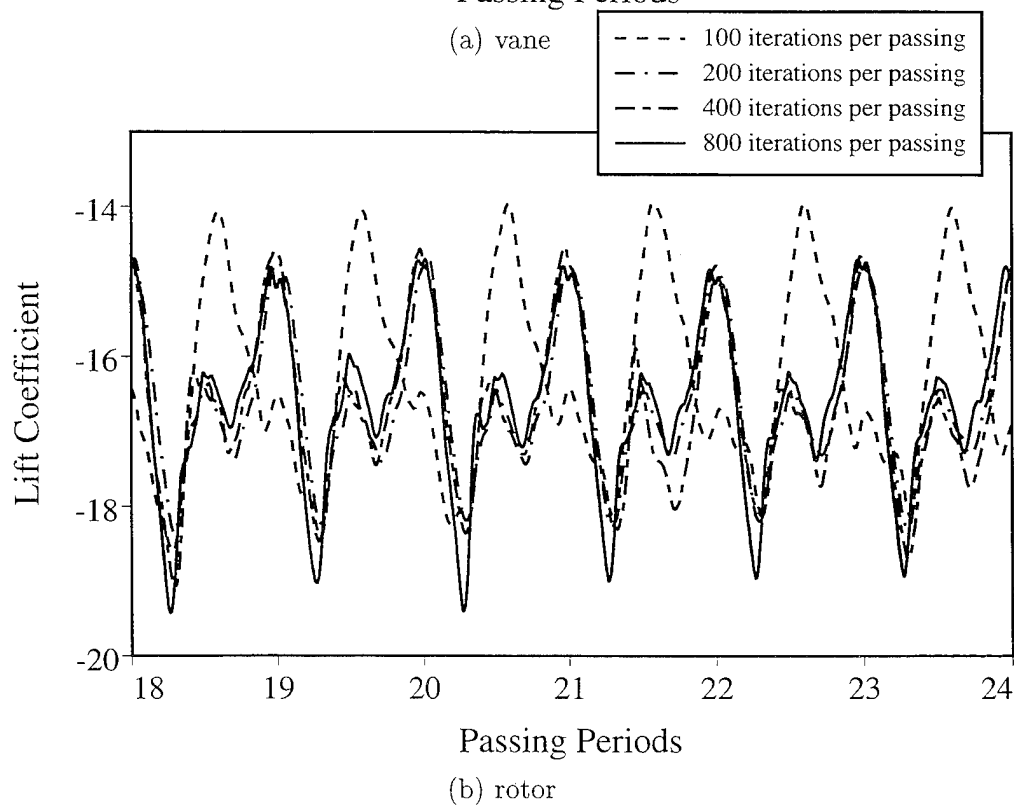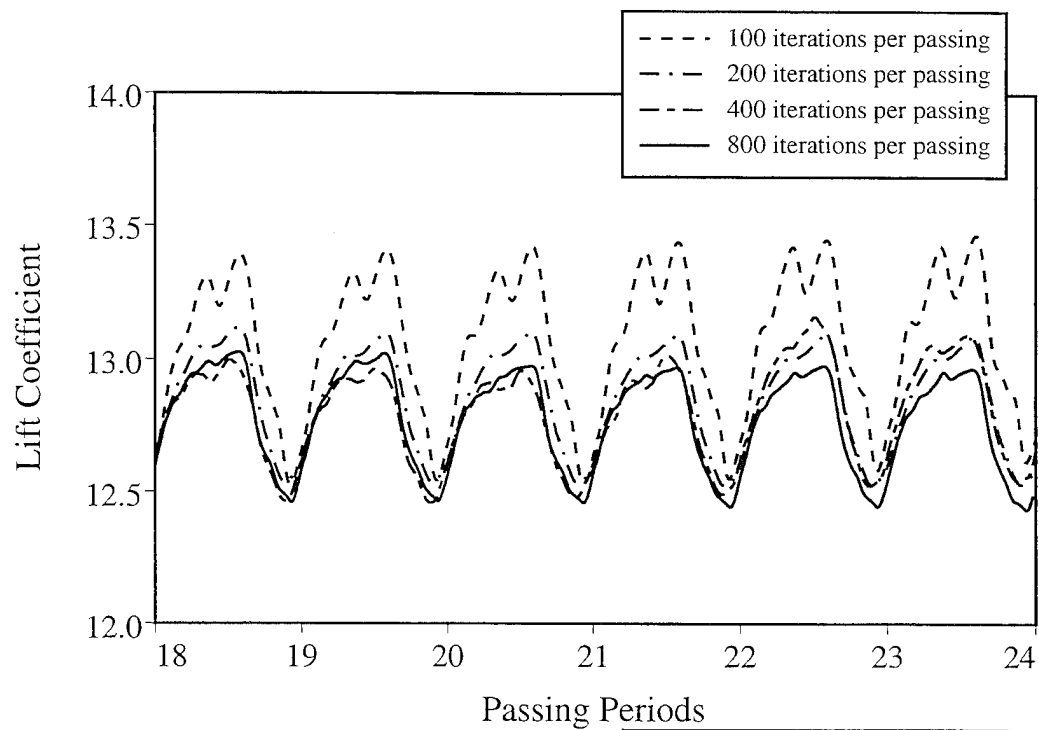
Figure 6.52: Variation in unsteady lift coefficient on adapted solution of turbine stage with different iteration counts.

(a) vane

(b) rotor
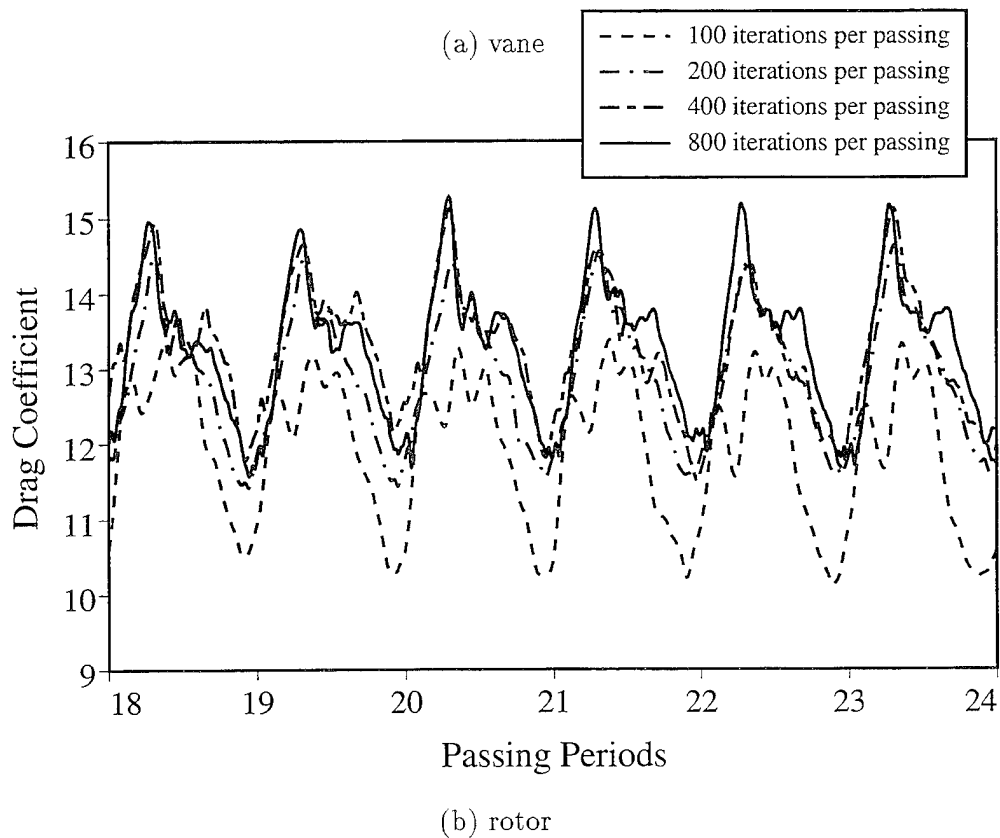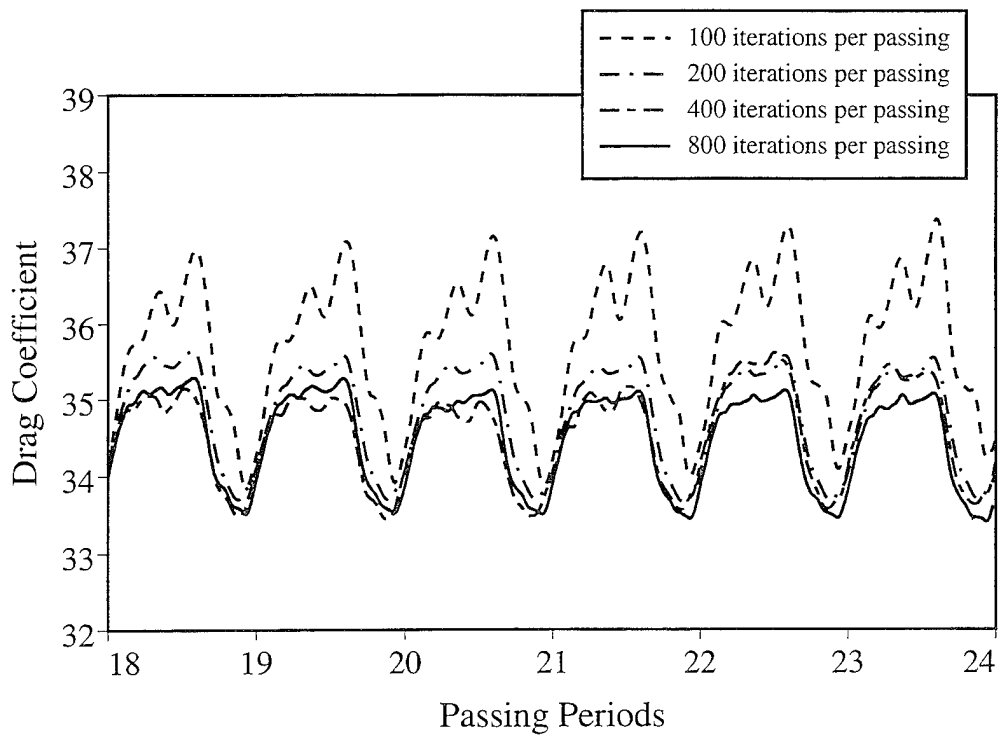
Figure 6.53: Variation in unsteady drag coefficient on adapted solution of turbine stage with different iteration counts.

Chapter 7

CONCLUSIONS

A solution-adaptive scheme for solving the quasi-three-dimensional Navier-Stokes equations through turbomachinery cascades has been described. The method uses central differencing of both the inviscid and viscous terms and a one-equation turbulence model for the Reynolds stresses. Time integration is performed in an implicit fashion through the use of a dual time stepping approach. The computational mesh is composed of micro-blocks of quadrilateral elements, arranged in a purely unstructured fashion. The mesh is adapted periodically as the solution evolves, using local cell division and recombination to enhance grid resolution at user-selected features in the flow field. This chapter summarizes the work and describes the conclusions that have been derived from it. Finally, a number of recommendations are presented for improvements to the code which might be explored in the future.

7.1 Summary

The aim of this effort was to produce a computational tool to permit the rapid computation of unsteady flows in modern transonic turbomachinery. The end product has shown the viability of using solution-adaptive, unstructured grid methods for unsteady flow computations. Considerable reduction in computer time is possible through the use of local cell division and recombination to distribute cells only where they can be the most useful. Computer time has been lowered further through the implementation of the dual time stepping approach, which allows an explicit time-

marching method to be used as a driver for a fully implicit scheme. This has allowed the use of convergence acceleration techniques, while maintaining a temporally accurate calculation.

7.2 Conclusions

The following section summarizes some of the major findings of this effort:

Dual Time Stepping Technique. The flow equations are recast in a manner which has allowed an explicit multi-stage Runge-Kutta scheme to be used as the driver for a fully implicit time integration method. The standard time derivative term in the flow equations is replaced with a second-order, backwards difference in time, which is treated as an additional source term. A pseudo-time variable is then introduced as a new integration variable in the Runge-Kutta scheme. Because the equations are no longer integrated in physical time, convergence techniques may be used; local time stepping and implicit residual smoothing have been implemented. The implicit formulation presented here is slightly different from the methods of Jameson [32] and Weiss and Smith [76] found in the literature, with a fully implicit treatment of the vector of dependent variables. This new scheme has a lower operation count than either method and does not require the limitation on the pseudo-time step size required by Jameson. A significant reduction in the number of iterations and computational time required for a converged periodic solution has been demonstrated as compared to a solution obtained with a purely explicit method.

Tests have been performed to evaluate the performance of the method in terms of the effect of the magnitude of the residual drop between increments in physical time and the effect of the number of real time steps per period. In these tests, the results have been shown to be relatively insensitive to the residual drop, with a large change in computer time found between the various cases. Results are considerably

160

more sensitive to the selection of the number of iterations used per period. For most solutions, a threshold value of iteration count exists above which only small improvements in the solution appear as the real time step size is reduced. Below this threshold, even the gross unsteady flow features are poorly predicted. The required iteration count per period is also shown to be strongly affected by the relative grid density. Both globally refined and adapted grids demonstrate a requirement for an increased number of iterations per period to obtain a temporally converged solution.

One-Equation Turbulence Model. The effects of turbulence have been modeled in the current effort with the one-equation model of Spalart and Allmaras [71]. Since the model contains both convective and diffusive terms, the full unsteady evolution of the turbulent eddy viscosity may be calculated and its effect on the blade row interaction evaluated. This model has been implemented with an integration scheme similar to that used for the flow equations and has proven to be quite robust. By including the turbulent eddy viscosity in the viscous time step limit used to compute the local pseudo-time step, source terms in the model may be evaluated explicitly and sub-iterations of the turbulence model are generally not required. The turbulence model has been shown to provide quite reasonable agreement with theoretical turbulent boundary layer velocity profiles and skin friction distributions, even for meshes with few points in the laminar sublayer. Compressor cascade computations show the prediction of premature separation on the blade suction surface and highlight the difficulty the model exhibits for cases with a large adverse pressure gradient. Blade row interactions within a turbine stage have been computed both with and without the turbulence model, as well as for the case where the turbulent eddy viscosity is not passed between the blade rows. A comparison of these cases has shown that the propagation of eddy viscosity generated by the upstream blade row has only a small effect on the computed lift and drag coefficient distributions.

161

Semi-Structured Grids. The semi-structured grid approach uses micro-blocks of structured cells arrayed in a globally unstructured fashion. The major benefits of this grid structure are the large reduction in the memory required to store the grid connectivity array and the lower computational time resulting from fewer indirect memory references. A primary reason for the selection of this type of grid structure was the speed at which grid adaptation could be performed. Because the logic surrounding the adaptation must only be applied to the various patches, rather than to each individual cell, the time required to divide or fuse a patch is greatly reduced; grid adaptation typically occurs in less than the time of a single pseudo-time iteration. Finally, the semi-structured approach should allow a natural implementation of the multigrid and parallel processing techniques described in the following section.

Blended O-H grids. The use of the blended O-H grids has proven quite successful in the computation of turbomachinery cascades, for both compressors and turbines. Many of the problems that occur when gridding blunt leading and trailing edges have been removed, while maintaining the flexibility of an H-grid structure in the main portion of the domain. Incorporation of the multi-block grid structure has further improved the grid through the removal of much of the grid skew that occurs at the point where more than two O- and H-grid blocks meet. Difficulties still exist in the generation of initial grids with extremely sheared grids, particularly for turbine cascades with their high turning angles.

Feature Detection. The adaptation of the computational grid is performed to locate various features within the solution flow field. Detection of these features is provided in two ways. A strong feature detector is used both to locate the regions of large pressure gradient found near shocks, and to avoid generation of additional grid refinement within the laminar sublayer of a turbulent velocity profile. The smooth feature detector, which finds regions of smooth, but rapid, changes in the flow

field, uses a statistical approach based on undivided differences of one or two user-selected flow parameters. Typically, Mach number and density are used as detection parameters, but turbine blade row interaction computations have shown that the turbulent eddy viscosity provides a superior method to track the propagation of the wake through the downstream blade row.

Smoothed Interpolation. When a micro-block of grid cells is divided, one method to locate the newly generated grid points is by simply averaging the locations of the adjacent points. An alternate method described here calculates the node locations based on the grid stretching currently existing with the patch. This method of cell division has proven quite effective in lowering the overall level of grid stretching error in the mesh. Reducing the stretching error is of particular importance in the highly stretched regions of the grid near blade surface, where accurate calculation of the viscous terms is critical. Maintaining the parent patch's grid line distribution in the divided patches allows the stretching ratio to be halved each time a patch is divided, further improving the accuracy of the solution.

7.3   Recommendations for Future Work

No CFD code can really ever reach the state where one might say that all the developmental work is truly "finished." While technology improves in all areas, progress in computational algorithms, turbulence models, and computing paradigms continues to drive the evolution of CFD tools. A number of these issues are discussed below:

Differencing of Convective Fluxes. The current implementation of the convective flux differencing uses a central differencing approach. While this method has a number of advantages, its primary disadvantage is the degree to which shocks are smeared, requiring second- and fourth-order damping. Monotonic methods,

which difference the fluxes along the characteristic directions in the flow, have a superior ability to resolve shocks, normally within one or two grid points. These methods could be implemented by forming the additional terms from the monotonic scheme as a replacement for the existing damping terms, while retaining the current central differencing of the inviscid fluxes.

Turbulence Model Improvements. The Spalart-Allmaras turbulence model has a number of advantages, both in its treatment of the turbulence physics and in its implementation. While this has made it attractive for this effort, it has been used primarily out of convenience. Little research has been conducted to date to construct turbulence models that are inexpensive to use and simple to implement across a wide range of unsteady flow problems. Most of the one- and two-equation models that are available for engineering use do well in low-speed, steady flows along flat plates in neutral pressure gradients. Real-world cases involve unsteady, transonic flows with a high degree of wall curvature and strong pressure gradients. It should be noted, however, that development of more advanced models is also driven by the availability of detailed test data in these more complex flow regimes; data acquisition and model development will advance hand-in-hand.

Extension to Three-Dimensions. While the use of the quasi-3D equations includes much of the flow physics found in modern turbomachinery designs, a number of effects are lost without a full 3-D implementation. These 3-D effects include shock sweep, endwall boundary layers, tip leakage, and other secondary flows. Extension of the semi-structured method to three dimensions should be fairly straight-forward and has already been accomplished, for steady flows, by Davis and Dannenhoffer [22].

Parallel Processing Implementation. The extension of the current work to a parallel processing environment would seem to be a natural progression. One of the primary difficulties of porting a computer code to either a fine- or coarse-

164

grained parallel processing environment is the proper partitioning of the solution domain for minimum communication overhead. The proper goal of any partitioning algorithm is to minimize the length of the boundary at the interface between partitions. The current implementation has an advantage over traditional unstructured codes in that it has already been partially partitioned into the individual patches. Partitioning the entire flow field would be done by combining all the patches within some level of parent patch. This should give the lowest partition boundary length because of the high perimeter-to-area ratio of the logically square patch shape. Once domain decomposition is accomplished, each partition's grid and flow variables are distributed to the various compute nodes, which then compute independently. Communication of the boundary data could occur at the end of each Runge-Kutta stage, or only after some fraction of the stages, with the boundary data frozen otherwise.

Multigrid. A significant advantage of the dual time stepping method is the ability to use techniques to accelerate solution convergence. Another method that might be implemented is multigrid, where a series of successively coarser grids is used to rapidly propagate changes in the convective fluxes throughout the domain. In the present semi-structured implementation, these coarse grids may be formed in one of two ways. The first method is to remove alternate grid lines within a patch for those patches at the finest level. Additional levels of grid coarseness may be obtained using the same method applied within the parents of divided patches. Multigrid is particularly useful for removing low frequency errors from the solution and therefore may become less effective as the size of the time step in physical time is reduced.

Appendix A

DERIVATION OF QUASI-3D EQUATIONS

This appendix presents the derivation of the quasi-3D form of the Navier-Stokes equations. As described in Chapter 2, these equations represent a two-dimensional formulation along a blade-to-blade streamsurface. The three-dimensional effects of streamsheet thickness variation and radius change are then added and appear as source terms in both the convective and diffusive terms of the equations.

Repeating the conservation equations given in three-dimensional form in Eqns. 2.1 - 2.5 :

$$\frac{d}{dt}\int\int\int_V \rho dV + \oint_{S_V} \rho \vec{V}_r \cdot \hat{n} dS = 0, \tag{A.1}$$

$$\frac{d}{dt}\int\int\int_V \rho \vec{V}_r dV + \oint_{S_V} \rho \vec{V}(\vec{V}_r \cdot \hat{n}) dS = \oint_{sV}(-p + \overline{\overline{\tau}}) \cdot \hat{n} dS, \tag{A.2}$$

$$\frac{d}{dt}\int\int\int_V E dV + \oint_{S_V} E(\vec{V}_r \cdot \hat{n}) dS = \dot{Q} - \oint_{S_V}(\overline{\overline{\tau}}\vec{V} - p\vec{V}) \cdot \hat{n} dS. \tag{A.3}$$

In order to expand the terms in these equations in the (m, θ) coordinate system shown in Fig. 2.2, vector identities are required for this system. Using the method of Anderson [3, pp. 193-197], with:

$$x_1 = r, \qquad h_1 = h,$$

$$x_2 = \theta, \qquad h_2 = r,$$

$$x_3 = m, \qquad h_3 = 1,$$

the following identities may be obtained (dropping the $\vec{A}_r$ terms since $v_r = 0$ in all cases):

$$\nabla \phi = \frac{1}{h}\frac{\partial \phi}{\partial r}\hat{\imath}_r + \frac{\partial \phi}{\partial m}\hat{\imath}_m + \frac{1}{r}\frac{\partial \phi}{\partial \theta}\hat{\imath}_\theta \tag{A.4}$$

$$\vec{\nabla} \cdot \vec{A} = \frac{1}{rh}\left[\frac{\partial}{\partial m}(rhA_m) + \frac{\partial}{\partial \theta}(hA_\theta)\right], \tag{A.5}$$

where ϕ is an arbitrary scalar and $\vec{A}$ is an arbitrary vector. The velocities $\vec{V}$ and $\vec{V}_r$ are expressed as:

$$\vec{V} = v_m\hat{\imath}_m + v_\theta\hat{\imath}_\theta, \tag{A.6}$$

$$\vec{V}_r = v_m\hat{\imath}_m + w_\theta\hat{\imath}_\theta, \tag{A.7}$$

where $w_\theta = v_\theta - r\Omega$.

Expressing the continuity equation in the equivalent vector differential form:

$$\frac{D\rho}{Dt} + \rho(\vec{\nabla} \cdot \vec{V}_r) = \frac{\partial \rho}{\partial t} + \vec{\nabla} \cdot (\rho\vec{V}_r) = 0. \tag{A.8}$$

Substituting $\vec{A} = \rho\vec{V}_r$ into Eqn. A.5, and bringing r and h inside the temporal derivative (since neither are functions of time):

$$\frac{\partial}{\partial t}(rh\rho) + \frac{\partial}{\partial m}(rh\rho v_m) + \frac{\partial}{\partial \theta}(h\rho w_\theta) = 0. \tag{A.9}$$

Similarly, the momentum equation may be expressed as:

$$\rho\frac{D\vec{V}}{Dt} = -\nabla p + \vec{\nabla} \cdot \bar{\bar{\tau}}. \tag{A.10}$$

168

Expanding the LHS of this equation and considering only the m and θ components yields:

$$\rho \frac{D\vec{V}}{Dt} = \rho \frac{\partial \vec{V}}{\partial t} + \rho \vec{V}_r \cdot \vec{\nabla} \vec{V}$$

$$= \rho \frac{\partial \vec{V}}{\partial t} + \rho \left(\frac{v_\theta}{r} \frac{\partial v_m}{\partial \theta} + v_m \frac{\partial v_m}{\partial m} - v_\theta w_\theta \frac{r_m}{r} \right) \hat{\imath}_m$$

$$+ \rho \left(\frac{v_\theta}{r} \frac{\partial w_\theta}{\partial \theta} + v_m \frac{\partial w_\theta}{\partial m} + v_\theta v_m \frac{r_m}{r} \right) \hat{\imath}_\theta, \qquad \text{(A.11)}$$

where the following notation is defined [15]:

$$\frac{r_m}{r} = \frac{1}{r} \frac{dr}{dm}, \qquad \frac{h_m}{h} = \frac{1}{h} \frac{dh}{dm}. \qquad \text{(A.12)}$$

Adding zero to both components of the equation in the form of the continuity equation and combining terms yields:

$$\rho \frac{D\vec{V}}{Dt} = \left(\frac{\partial}{\partial t}(rh\rho v_m) + \frac{\partial}{\partial m}(rh\rho v_m^2) + \frac{\partial}{\partial \theta}(h\rho v_m w_\theta) - \rho v_\theta^2 \frac{r_m}{r} \right) \frac{\hat{\imath}_m}{rh}$$

$$+ \left(\frac{\partial}{\partial t}(r^2 h\rho v_\theta) + \frac{\partial}{\partial m}(r^2 h\rho v_m v_\theta) + \frac{\partial}{\partial \theta}(rh\rho w_\theta v_\theta) - \rho v_\theta v_m \frac{r_m}{r} \right) \frac{\hat{\imath}_\theta}{rh}. $$

$$\text{(A.13)}$$

The RHS of Eqn. A.10 is also expanded using the method outlined in Anderson [3, pg. 196]. The components of the total stress tensor $\overline{\overline{\Pi}} = -p\delta_{ij} + \overline{\overline{\tau}}$ are given by:

$$\Pi_{rr} = -p + \frac{2}{3}\mu \left(2e_{rr} - e_{\theta\theta} - e_{mm} \right),$$

$$\Pi_{\theta\theta} = -p + \frac{2}{3}\mu \left(2e_{\theta\theta} - e_{mm} - e_{rr} \right),$$

$$\Pi_{mm} = -p + \frac{2}{3}\mu \left(2e_{mm} - e_{rr} - e_{\theta\theta} \right), \qquad \text{(A.14)}$$

169

$$\Pi_{r\theta} = \Pi_{\theta r} = \mu e_{r\theta},$$

$$\Pi_{\theta m} = \Pi_{m\theta} = \mu e_{\theta m},$$

$$\Pi_{mr} = \Pi_{rm} = \mu e_{mr}.$$

The expressions for the strain components in the stress tensor are:

$$
\begin{aligned}
e_{rr} &= v_m \frac{h_m}{h}, \\
e_{\theta\theta} &= \frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m \frac{r_m}{r}, \\
e_{mm} &= \frac{\partial v_m}{\partial m}, \\
e_{r\theta} &= \frac{r}{h}\frac{\partial}{\partial r}\left(\frac{v_\theta}{r}\right), \\
e_{\theta m} &= \frac{1}{r}\frac{\partial v_m}{\partial \theta} + r\frac{\partial}{\partial m}\left(\frac{v_\theta}{r}\right), \\
e_{mr} &= \frac{1}{h}\frac{\partial v_m}{\partial r}.
\end{aligned}
\tag{A.15}
$$

The components of the total stress tensor are then given by:

$$
\begin{aligned}
\Pi_{rr} &= -p + \frac{2}{3}\mu\left[2v_m\frac{h_m}{h} - \left(\frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m\frac{r_m}{r}\right) - \frac{\partial v_m}{\partial m}\right], \\
\Pi_{\theta\theta} &= -p + \frac{2}{3}\mu\left[2\left(\frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m\frac{r_m}{r}\right) - v_m\frac{h_m}{h} - \frac{\partial v_m}{\partial m}\right], \\
\Pi_{mm} &= -p + \frac{2}{3}\mu\left[2\frac{\partial v_m}{\partial m} - v_m\frac{h_m}{h}\left(\frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m\frac{r_m}{r}\right)\right], \\
\Pi_{r\theta} &= \mu\left[\frac{r}{h}\frac{\partial}{\partial r}\left(\frac{v_\theta}{r}\right)\right], \\
\Pi_{\theta m} &= \mu\left[\frac{1}{r}\frac{\partial v_m}{\partial \theta} + r\frac{\partial}{\partial m}\left(\frac{v_\theta}{r}\right)\right], \\
\Pi_{mr} &= \mu\left[\frac{1}{h}\frac{\partial v_m}{\partial r}\right].
\end{aligned}
\tag{A.16}
$$

Finally, taking $\vec{\nabla} \cdot \overline{\overline{\Pi}}$ and rearranging yields:

$$
\begin{aligned}
\vec{\nabla} \cdot \overline{\overline{\Pi}} &= -\nabla p + \vec{\nabla} \cdot \overline{\overline{\tau}} \\
&= \left(\frac{\partial}{\partial m}(rh(-p + \tau_{mm})) + \frac{\partial}{\partial \theta}(r\tau_{m\theta}) + (p - \tau_{\theta\theta})\frac{r_m}{r} + (p - \tau_{rr})\frac{h_m}{h} \right) \frac{\hat{i}_m}{rh} \\
&+ \left(\frac{\partial}{\partial m}(r^2 h \tau_{m\theta}) + \frac{\partial}{\partial \theta}(rh(-p + \tau_{\theta\theta})) \right) \frac{\hat{i}_\theta}{rh};
\end{aligned} \tag{A.17}
$$

where

$$
\begin{aligned}
\tau_{rr} &= 2\mu v_m \frac{h_m}{h} - \frac{2}{3}\mu\Theta, \\
\tau_{\theta\theta} &= 2\mu \left(\frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m \frac{r_m}{r} \right) - \frac{2}{3}\mu\Theta, \\
\tau_{mm} &= 2\mu \frac{\partial v_m}{\partial m} - \frac{2}{3}\mu\Theta, \\
\tau_{m\theta} &= \mu(\frac{\partial v_\theta}{\partial m} + \frac{1}{r}\frac{\partial v_m}{\partial \theta} - v_\theta \frac{r_m}{r}),
\end{aligned} \tag{A.18}
$$

and the dilatation is given by:

$$
\Theta = \left(\frac{\partial v_m}{\partial m} + \frac{1}{r}\frac{\partial v_\theta}{\partial \theta} + v_m \left(\frac{r_m}{r} + \frac{h_m}{h} \right) \right). \tag{A.19}
$$

Combining Eqns. A.13 and A.17 gives the two momentum equations:

$$
\begin{aligned}
\frac{\partial}{\partial t}(rh\rho v_m) &+ \frac{\partial}{\partial m}(rh(\rho v_m^2 + p - \tau_{mm})) + \frac{\partial}{\partial \theta}(h(\rho v_m w_\theta - \tau_{m\theta})) = \\
&= rh \left((\rho v_\theta^2 + p - \tau_{\theta\theta})\frac{r_m}{r} + (p - \tau_{rr})\frac{h_m}{h} \right),
\end{aligned} \tag{A.20}
$$

$$
\begin{aligned}
\frac{\partial}{\partial t}(rh\rho v_\theta) &+ \frac{\partial}{\partial m}(rh(\rho v_m v_\theta - \tau_{m\theta})) + \frac{\partial}{\partial \theta}(h(\rho w_\theta v_\theta + p - \tau_{\theta\theta})) = \\
&= -rh \left(\rho v_\theta v_m - \tau_{m\theta} \right) \frac{r_m}{r}.
\end{aligned} \tag{A.21}
$$

171

By multiplying both sides of the θ-direction momentum equation by r, and using the chain rule, the source term on the RHS of the equation may be absorbed to give:

$$\frac{\partial}{\partial t}(r^2 h\rho v_\theta) + \frac{\partial}{\partial m}(r^2 h(\rho v_m v_\theta - \tau_{m\theta})) + \frac{\partial}{\partial \theta}(rh(\rho w_\theta v_\theta + p - \tau_{\theta\theta})) = 0. \qquad (A.22)$$

The energy equation may be expressed in vector differential form as:

$$\frac{\partial E}{\partial t} + \vec{\nabla} \cdot (E\vec{V}_r) = -\vec{\nabla} \cdot \vec{q} + \vec{\nabla} \cdot (\overline{\overline{\Pi}} \cdot \vec{V}). \qquad (A.23)$$

Using the same technique as used in the continuity equation, the LHS is expanded to yield:

$$\frac{\partial E}{\partial t} + \vec{\nabla} \cdot (E\vec{V}_r) = \frac{1}{rh}\left(\frac{\partial E}{\partial t} + \frac{\partial}{\partial m}(rhEv_m) + \frac{\partial}{\partial \theta}(hEw_\theta)\right). \qquad (A.24)$$

Using Fourier's Law, $\vec{q} = -\kappa\nabla T$, and neglecting temperature gradients in the radial direction, Eqns. A.4 and A.5 give:

$$-\vec{\nabla} \cdot \vec{q} = \frac{1}{rh}\left(\frac{\partial}{\partial m}(rh\kappa\frac{\partial T}{\partial m}) + \frac{\partial}{\partial \theta}(h\kappa\frac{1}{r}\frac{\partial T}{\partial \theta})\right). \qquad (A.25)$$

Assuming that the radial shear stresses Π_{rm} and $\Pi_{r\theta}$ are zero yields:

$$\begin{aligned}
\overline{\overline{\Pi}} \cdot \vec{V} &= \vec{A} \\
&= (v_m(-p + \tau_{mm}) + v_\theta\tau_{m\theta})\,\hat{i}_m \\
&\quad + (v_m\tau_{m\theta} + v_\theta(-p + \tau_{\theta\theta}))\,\hat{i}_\theta, \qquad (A.26)
\end{aligned}$$

and

$$\vec{\nabla} \cdot (\overline{\overline{\Pi}} \cdot \vec{V}) = \frac{1}{rh}\left(\frac{\partial}{\partial m}(rhA_m) + \frac{\partial}{\partial \theta}(hA_\theta)\right). \qquad (A.27)$$

Combining all terms and simplifying, the final form of the energy equation is obtained:

$$\frac{\partial}{\partial t}(rhE) + \frac{\partial}{\partial m}\left(rh(E+p)v_m - R_4)\right)$$
$$+ \frac{\partial}{\partial \theta}\left(h(E+p)w_\theta - S_4 + r\Omega p)\right) = 0, \qquad (A.28)$$

where

$$R_4 = v_m\tau_{mm} + v_\theta\tau_{m\theta} + \kappa\frac{\partial T}{\partial m}, \qquad (A.29)$$

$$S_4 = v_m\tau_{m\theta} + v_\theta\tau_{\theta\theta} + \kappa\frac{1}{r}\frac{\partial T}{\partial \theta}. \qquad (A.30)$$

Finally, the continuity, momentum, and energy equations may be expressed in their final vector form:

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial m} + \frac{\partial G}{\partial \theta} = \frac{\partial R}{\partial m} + \frac{\partial S}{\partial \theta} + H, \qquad (A.31)$$

where

$$U = rh\begin{bmatrix} \rho \\ \rho v_m \\ \rho v_\theta r \\ E \end{bmatrix}, \quad F = rh\begin{bmatrix} \rho v_m \\ \rho v_m^2 + p \\ (\rho v_m v_\theta)r \\ v_m(E+p) \end{bmatrix},$$

$$G = h\begin{bmatrix} \rho w_\theta \\ \rho w_\theta v_m \\ (\rho w_\theta v_\theta + p)r \\ w_\theta(E+p) + r\Omega p \end{bmatrix}, \quad H = rh\begin{bmatrix} 0 \\ H_2 \\ 0 \\ 0 \end{bmatrix}, \qquad (A.32)$$

$$R = rh \begin{bmatrix} 0 \\ \tau_{mm} \\ \tau_{m\theta} r \\ R_4 \end{bmatrix}, \quad S = h \begin{bmatrix} 0 \\ \tau_{m\theta} \\ \tau_{\theta\theta} r \\ S_4 \end{bmatrix},$$

where the source term is:

$$H_2 = (\rho v_\theta^2 + p - \tau_{\theta\theta}) \frac{r_m}{r} + (p - \tau_{rr}) \frac{h_m}{h}. \qquad \text{(A.33)}$$

Appendix B

CHARACTERISTIC BOUNDARY CONDITIONS

The method for specifying the inflow and outflow boundary conditions is based on the development of non-reflecting conditions for the Euler equations described by Giles [27, 28]. By using characteristic variables to specify the dependent variable values at the boundaries, the pressure waves generated by the initial solution transients are allowed to exit the domain without reflection. This results in faster convergence, an ability to place computational boundaries closer to the solid bodies, and a greater tolerance of the flow solver to poorly specified initial conditions. This appendix discusses the theory behind these boundary conditions and describes some of the special considerations required by turbomachinery problems.

B.1 Review of Characteristic Theory

Development of characteristic theory is based on a consideration of the Euler equations written in the conservation form described in the previous Appendix. However, the development in this Appendix will neglect the change in radius and streamsheet thickness. The assumption is made that the inflow and outflow boundaries occur in regions of nearly constant radius and streamsheet thickness, reasonable assumptions for axial turbomachinery, but invalid for centrifugal machines.

The Euler equations may be transformed into an equivalent form based on the primitive variables ρ, v_m, v_θ, p. These equations are linearized by considering only small perturbations and neglecting the higher order terms, yielding the following

form:

$$
\frac{\partial}{\partial t}\begin{pmatrix} \tilde{\rho} \\ \tilde{v}_m \\ \tilde{v}_\theta \\ \tilde{p} \end{pmatrix} + \begin{pmatrix} v_m & \rho & 0 & 0 \\ 0 & v_m & 0 & \frac{1}{\rho} \\ 0 & 0 & v_m & 0 \\ 0 & \gamma p & 0 & v_m \end{pmatrix} \frac{\partial}{\partial m}\begin{pmatrix} \tilde{\rho} \\ \tilde{v}_m \\ \tilde{v}_\theta \\ \tilde{p} \end{pmatrix} + \begin{pmatrix} v_\theta & 0 & \rho & 0 \\ 0 & v_\theta & 0 & 0 \\ 0 & 0 & v_\theta & \frac{1}{\rho} \\ 0 & 0 & \gamma p & v_\theta \end{pmatrix} \frac{1}{r}\frac{\partial}{\partial \theta}\begin{pmatrix} \tilde{\rho} \\ \tilde{v}_m \\ \tilde{v}_\theta \\ \tilde{p} \end{pmatrix} = 0,
$$

(B.1)

where $\tilde{\rho}, \tilde{v}_m, \tilde{v}_\theta, \tilde{p}$ are the perturbation quantities about the uniform flow values ρ, v_m, v_θ, p. For inflow and outflow boundaries, only axial variations are considered and therefore changes in the circumferential direction are neglected.

Performing a standard eigenvalue analysis, the characteristic variables of the resulting one-dimensional equation are given by [27]:

$$
\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix} = \begin{pmatrix} -c^2 & 0 & 0 & 1 \\ 0 & 0 & \rho c & 0 \\ 0 & \rho c & 0 & 1 \\ 0 & -\rho c & 0 & 1 \end{pmatrix} \begin{pmatrix} \tilde{\rho} \\ \tilde{v}_m \\ \tilde{v}_\theta \\ \tilde{p} \end{pmatrix}.
$$

(B.2)

The characteristic variables c_1, c_2, c_3, c_4 represent an entropy wave, a vorticity wave, and downward- and upward-running pressure waves respectively; c is the isentropic sound speed. The c_1, c_2, c_3, c_4 values are the amplitude of these waves, and $\tilde{\rho}$, $\tilde{v}_m$, $\tilde{v}_\theta$, $\tilde{p}$ are the perturbations from the uniform flow used for the linearization.

This relation is used to develop boundary conditions by implementing the conditions as modifications to the flux variables at the boundaries, rather than as a specification of the actual boundary values. For an inflow boundary, either three or four of the characteristic variables must be specified, depending upon the inflow Mach number. At the outflow boundary, the specification of one characteristic may be required, again depending upon the Mach number. Characteristic variables that

are not specified are extrapolated from within the flow domain: the mechanism which allows pressure waves to exit the domain at either boundary.

If the change in the primitive variables is known at some point, then the change in the characteristic variables may be found using the a relation similar to Eqn. B.2:

$$
\begin{pmatrix} \delta c_1 \\ \delta c_2 \\ \delta c_3 \\ \delta c_4 \end{pmatrix} = \begin{pmatrix} -c^2 & 0 & 0 & 1 \\ 0 & 0 & \rho c & 0 \\ 0 & \rho c & 0 & 1 \\ 0 & -\rho c & 0 & 1 \end{pmatrix} \begin{pmatrix} \delta \rho \\ \delta v_m \\ \delta v_\theta \\ \delta p \end{pmatrix},
\tag{B.3}
$$

with the inverse given by:

$$
\begin{pmatrix} \delta \rho \\ \delta v_m \\ \delta v_\theta \\ \delta p \end{pmatrix} = \begin{pmatrix} -\frac{1}{c^2} & 0 & \frac{1}{2c^2} & \frac{1}{2c^2} \\ 0 & 0 & \frac{1}{2\rho c} & -\frac{1}{2\rho c} \\ 0 & \frac{1}{\rho c} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} \delta c_1 \\ \delta c_2 \\ \delta c_3 \\ \delta c_4 \end{pmatrix}.
\tag{B.4}
$$

Here, $\delta c_1, \delta c_2, \delta c_3, \delta c_4$ are the calculated characteristic variable amplitudes and $\delta \rho$, δv_m, δv_θ, δp are the actual changes in the primitive variables from which the changes in the dependent variables may be derived.

The basic technique in formulating these boundary conditions is to first choose an appropriate residual to be reduced, then find the resultant changes in the characteristic variables required to achieve this reduction. Changes to the dependent variable values at the boundaries are then calculated using the relations described above. These residuals are based on the user-specified flow conditions for the particular case to be computed and must be chosen with regard to the type of flow (subsonic or supersonic, inflow or outflow) that is expected at the boundary. The

following sections describe these various choices and their respective implementations.

B.2 Inlet Boundary Conditions

At the inflow boundary, at least three of the four characteristics are incoming, with the direction of the fourth determined by the inflow axial Mach number. In the special case of a supersonic inlet relative flow with a subsonic axial Mach number, which is typical of transonic compressors, additional constraints are placed on the boundary condition by the interaction between the inlet flow and the geometry of the cascade. The following section discusses the implementation of these various possibilities.

B.2.1 Subsonic Inlet Flow

For a subsonic inlet relative flow, the inlet flow is purely subsonic and a fairly standard boundary condition may be implemented. In this case the residuals to be driven to zero are based on the inlet entropy, inlet flow angle, and stagnation enthalpy:

$$
\begin{aligned}
R_1 &= p(S - S_\infty), \\
R_2 &= \rho c(v_\theta - \tan(\alpha_\infty)v_m), \\
R_3 &= \rho(H - H_\infty), \quad\quad\quad\quad\quad\quad\quad\quad\quad\text{(B.5)}
\end{aligned}
$$

where S is an entropy function given by:

$$
S = \log(\gamma p) - \gamma \log(\rho), \quad\quad\quad\quad\quad\quad\text{(B.6)}
$$

and H is the stagnation enthalpy:

$$H = E + p. \tag{B.7}$$

Nondimensionalizing by the inlet velocity and static density, the specified values of these functions become:

$$
\begin{aligned}
S_\infty &= \log\left(\frac{1}{M_\infty^2}\right), \\
H_\infty &= \frac{1}{(\gamma - 1)M_\infty^2} + \frac{1}{2}.
\end{aligned}
$$

Following the method of Giles [28], a single iteration of a Newton-Ralphson procedure is performed to determine the changes required in the characteristic variables to drive these residuals to zero:

$$
\begin{pmatrix} R_1 \\ R_2 \\ R_3 \end{pmatrix}^n + \frac{\partial(R_1, R_2, R_3)}{\partial(c_1, c_2, c_3)} \begin{pmatrix} \delta\bar{c}_1 \\ \delta\bar{c}_2 \\ \delta\bar{c}_3 \end{pmatrix} = 0, \tag{B.8}
$$

where the following expressions may be obtained:

$$
\begin{aligned}
\frac{\partial(R_1, R_2, R_3)}{\partial(c_1, c_2, c_3)} &= \frac{\partial(R_1, R_2, R_3)}{\partial(\rho, v_m, v_\theta, p)} \frac{\partial(\rho, v_m, v_\theta, p)}{\partial(c_1, c_2, c_3)} \\
&= \begin{pmatrix} -c^2 & 0 & 0 & 1 \\ 0 & \rho c \tan(\alpha_\infty) & \rho c & 0 \\ -\frac{1}{\gamma-1}c^2 & \rho v_m & \rho v_\theta & \frac{\gamma}{\gamma-1} \end{pmatrix} \begin{pmatrix} -\frac{1}{c^2} & 0 & \frac{1}{2c^2} \\ 0 & 0 & \frac{1}{2\rho c} \\ 0 & \frac{1}{\rho c} & 0 \\ 0 & 0 & \frac{1}{2} \end{pmatrix}
\end{aligned}
$$

179

$$
= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -\frac{1}{2}\tan(\alpha_\infty) \\ \frac{1}{\gamma-1} & M_\theta & \frac{1}{2}(1+M_m) \end{pmatrix}.
\tag{B.9}
$$

In the above relation, M_m and M_θ are the Mach numbers in the m and θ directions respectively. Inverting this matrix, the required changes in the characteristic variables are obtained:

$$
\begin{pmatrix} \delta\bar{c}_1 \\ \delta\bar{c}_2 \\ \delta\bar{c}_3 \end{pmatrix} = \frac{-1}{1+M_m+M_\theta\tan(\alpha_\infty)} \times
$$

$$
\begin{pmatrix} 1+M_m+M_\theta\tan(\alpha_\infty) & 0 & 0 \\ \frac{1}{\gamma-1}\tan(\alpha_\infty) & 1+M_m & \tan(\alpha_\infty) \\ -\frac{1}{\gamma-1} & -2M_\theta & 2 \end{pmatrix} \begin{pmatrix} R_1 \\ R_2 \\ R_3 \end{pmatrix}. \tag{B.10}
$$

The change in the fourth (outgoing) characteristic variable is determined by extrapolating its value from a point adjacent to the boundary, using the values computed during the inviscid flux integration:

$$
\delta\bar{c}_4 = -\rho c\,\delta v_m + \delta p,
\tag{B.11}
$$

where

$$
\delta\rho = \delta U_1,
$$

$$
\delta v_m = (\delta U_2 - v_m\delta U_1)/\rho,
$$

$$
\delta v_\theta = (\delta U_3 - v_\theta\delta U_1)/\rho,
$$

$$
\delta p = (\gamma-1)\left(\delta U_4 - v_m\delta U_2 - v_\theta\delta U_3 + \frac{1}{2}(v_m^2+v_\theta^2)\delta U_1\right).
\tag{B.12}
$$

Here the δU values are evaluated at the next point away from the inlet boundary. The changes in the primitive variables may be obtained by applying Eqn. B.4:

$$\begin{aligned}
\delta\rho &= \frac{1}{c^2}\left[-\delta\bar{c}_1 + \frac{1}{2}(\delta\bar{c}_3 + \delta\bar{c}_4)\right], \\
\delta v_m &= \frac{1}{2\rho c}(\delta\bar{c}_3 - \delta\bar{c}_4), \\
\delta v_\theta &= \frac{\delta\bar{c}_2}{\rho c}, \\
\delta p &= \frac{1}{2}(\delta\bar{c}_3 + \delta\bar{c}_4).
\end{aligned} \tag{B.13}$$

Finally, from these values, the required changes in the dependent variables at the inlet boundary are found:

$$\begin{aligned}
\delta U_1 &= \delta\rho, \\
\delta U_2 &= \rho\delta v_m + v_m\delta\rho, \\
\delta U_3 &= \rho\delta v_\theta + v_\theta\delta\rho, \\
\delta U_4 &= \frac{\delta p}{\gamma - 1} + v_m\delta U_2 + v_\theta\delta U_3 - \frac{1}{2}(v_m^2 + v_\theta^2)\delta U_1.
\end{aligned} \tag{B.14}$$

The radius and streamsheet contraction terms must also be included in these dependent variable changes; it is also important to remove these terms when extrapolating $\delta\bar{c}_4$ in Eqn. B.11.

B.2.2 Supersonic Inlet Flow

For a supersonic inlet flow condition, with the axial Mach number still subsonic, an alteration of the inlet boundary condition is required. In a transonic compressor cascade, it is the interaction between the inlet flow and the geometry of the cascade which sets the inlet flow angle; the mass flow is set by the exit static pressure. This is referred to as the *unique incidence condition* [18]. Because of this condition, the

inlet flow angle can no longer be specified; instead, the inlet relative Mach number or the tangential velocity (wheel speed) must be given. The inlet flow angle is then determined after the cascade flow has converged. This condition is also the reason why a transonic fan has such a small variation in mass flow over its operating characteristic; there is only a small range of acceptable inlet flow angles between the point at which the cascade chokes and where the bow shock spills, causing the cascade to stall. Many researchers, including Giles, miss this point and incorrectly specify the inlet flow angle. Use of this condition may yield a convergent solution, however it does deny some of the important physical phenomena at work.

The specification of the supersonic inlet boundary condition proceeds in a fashion similar to that developed for the subsonic condition. In this case, the residuals to be driven to zero are based on the inlet entropy, tangential velocity, and stagnation enthalpy:

$$
\begin{aligned}
R_1 &= p(S - S_\infty), \\
R_2 &= \rho c(v_\theta - v_{\theta_\infty}), \\
R_3 &= \rho(H - H_\infty).
\end{aligned}
\tag{B.15}
$$

The procedure outlined previously is used, yielding the following differences from the subsonic results:

$$
\frac{\partial(R_1, R_2, R_3)}{\partial(c_1, c_2, c_3)} =
\begin{pmatrix}
-c^2 & 0 & 0 & 1 \\
0 & 0 & \rho c & 0 \\
-\frac{1}{\gamma-1}c^2 & \rho v_m & \rho v_\theta & \frac{\gamma}{\gamma-1}
\end{pmatrix}
\begin{pmatrix}
-\frac{1}{c^2} & 0 & \frac{1}{2c^2} \\
0 & 0 & \frac{1}{2\rho c} \\
0 & \frac{1}{\rho c} & 0 \\
0 & 0 & \frac{1}{2}
\end{pmatrix}
$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{1}{\gamma-1} & M_\theta & \frac{1}{2}(1 + M_m) \end{pmatrix}. \tag{B.16}$$

Inverting this matrix gives the required change in the characteristic variables:

$$\begin{pmatrix} \delta\bar{c}_1 \\ \delta\bar{c}_2 \\ \delta\bar{c}_3 \end{pmatrix} = \frac{-1}{1 + M_m} \begin{pmatrix} 1 + M_m & 0 & 0 \\ 0 & 1 + M_m & 0 \\ -\frac{1}{\gamma-1} & -2M_\theta & 2 \end{pmatrix} \begin{pmatrix} R_1 \\ R_2 \\ R_3 \end{pmatrix} \tag{B.17}$$

The boundary condition specification is completed using a procedure identical to that used for the subsonic condition.

B.2.3 Pure Supersonic Inflow

For an inlet where the flow is purely supersonic, i.e. has a supersonic axial Mach number, all four characteristic variables must be specified. Assuming that this is also a steady flow specification, there will be no change in the characteristic variables from the values used to originally specify the initial conditions of the problem. Therefore, the inlet condition is quite simple:

$$\delta\bar{c}_1 = \delta\bar{c}_2 = \delta\bar{c}_3 = \delta\bar{c}_4 = 0, \tag{B.18}$$

or equivalently:

$$\delta U_1 = \delta U_2 = \delta U_3 = \delta U_4 = 0. \tag{B.19}$$

This is the boundary condition formulation used in the supersonic flat plate boundary layer cases described previously.

B.3 Outflow Conditions

At the outflow boundary, at least three of the four characteristics are outgoing, with the direction of the fourth determined by the exit axial Mach number. The following sections describe these two possibilities:

B.3.1 Supersonic Outflow

If the exit axial Mach number is supersonic, then the fourth characteristic is also outgoing, and no specification of the exit quantities is allowed. The changes in the flow variables at the exit boundary are found by a simple extrapolation from the fluxes calculated at the adjacent grid point:

$$
\begin{aligned}
\delta U_{1_{exit}} &= \delta U_{1_{exit-}}, \\
\delta U_{2_{exit}} &= \delta U_{2_{exit-}}, \\
\delta U_{3_{exit}} &= \delta U_{3_{exit-}}, \\
\delta U_{4_{exit}} &= \delta U_{4_{exit-}}.
\end{aligned}
\tag{B.20}
$$

The subscript *exit* indicates points along the exit boundary, and *exit⁻* indicates points just in from the boundary. Again, it should be noted that the ratios of the streamsheet contraction and radius must be included in this evaluation.

B.3.2 Subsonic Outflow

For a subsonic exit Mach number condition, the downstream static pressure is typically specified. This is equivalent to the setting the cascade pressure ratio, often prescribed in compressor and turbine cascade calculations. In this case, the fourth characteristic variable is incoming and must therefore be determined. Here, the

residual to be driven to zero is based on the difference between the actual and specified static pressure. The fourth characteristic variable is given by:

$$\delta \bar{c}_4 = -2(p - p_{exit}), \tag{B.21}$$

where p_{exit} is the user specified value.

The other three characteristic variable values are determined using the simple extrapolation described above to give:

$$
\begin{aligned}
\delta \bar{c}_{1_{exit}} &= \delta \bar{c}_{1_{exit-}}, \\
\delta \bar{c}_{2_{exit}} &= \delta \bar{c}_{2_{exit-}}, \\
\delta \bar{c}_{3_{exit}} &= \delta \bar{c}_{3_{exit-}}.
\end{aligned}
\tag{B.22}
$$

Having the values for the change in all four characteristics at the exit boundary, the changes in the dependent variables may be calculated as before.

B.3.3 Mixed Exit Condition

A mixed subsonic-supersonic exit condition has also been implemented for use in the supersonic boundary layer calculations used for flow solver validation. In this case, a search out from the wall along the exit boundary is performed to locate the first point at which the flow becomes supersonic. The supersonic points are treated with the extrapolation condition, while subsonic points use the pressure at the first supersonic point to set their exit pressure.

Appendix C

INTERACTIVE GRID GENERATOR

This appendix describes the grid generator used to provide the initial computational grids for cascade calculations. As discussed in Chapter 5, the grids produced are in a multi-block format and include both single and multiple blade row cases. Additionally, the grid generator is responsible for the creation of the streamsheet thickness and radius variation data files. The gridding tool uses a graphical user interface (GUI) which allows the user to access virtually all of the parameters needed to generate a multi-block grid. The following sections describe the geometrical data requirements, the multi-block grid format, the elliptic and algebraic grid generation algorithms, and the graphical interface. This appendix serves as a brief introduction to the grid generator and is not intended as an user's guide, this will be left to a later publication.

C.1 Multi-Block Grid Structure

The grids constructed for this thesis use a multi-block O-H grid structure decomposition of the computational domain. Using the combination of these two grid types allows the user to take advantage of the relative advantages of each type. The topological decomposition of a typical grid is shown in Fig. C.1. An O-grid is wrapped around the blade to use its superior ability to produce grid lines which are nearly normal to the blade surface. Further, O-grids allow leading and trailing edges with small radii to be gridded with minimal grid skew. The O-grid is split into two halves,

Figure C.1: Logical topology for multi-block grids.

divided along the periodic boundaries which extend from the leading and trailing edges. These two blade grids are blended into a passage grid which is composed of multiple H-grid blocks. These blocks consist of a main passage grid (labeled A in Fig. C.1) and optional blocks upstream (blocks B and D) and downstream (blocks C and E) of the two O-grid halves. During the grid generation procedure described below, the user has control over the number of grid lines in each of these blocks. The H-grid structure was selected for the passage grid to facilitate the construction of the inlet, exit, and sliding interface boundary conditions, which occur along the right and left edges of the H-grid block. Periodic boundaries in the H-grid portion occur between the bottom of block B and top of block D, and similarly between blocks C and E. All H-grid blocks are stored together in a global H-grid domain with counters used to locate the corners of the various blocks. These counters are written to a separate file used by the flow solver to reconstruct the H-grid blocks and position the O-grid halves.

C.2   Geometry Definition Requirements

In order to facilitate the integration of the computational tool described in this thesis into the design system used in the US Air Force Wright Laboratory, Aero Propulsion and Power Directorate, a standard data format was adopted. This format was taken from the geometry definition files generated by the in-house fan and compressor design program UD0300. This program uses a streamline curvature method to develop blade shapes from user input performance parameters; a limited description of this program is available in [40, 77].

Two types of geometrical data files are used by the grid generator in the creation of a quasi-3D blade-to-blade grid surface. The first is a definition of the blade shape on a series of streamsurfaces ordered from hub to tip. These shapes are true
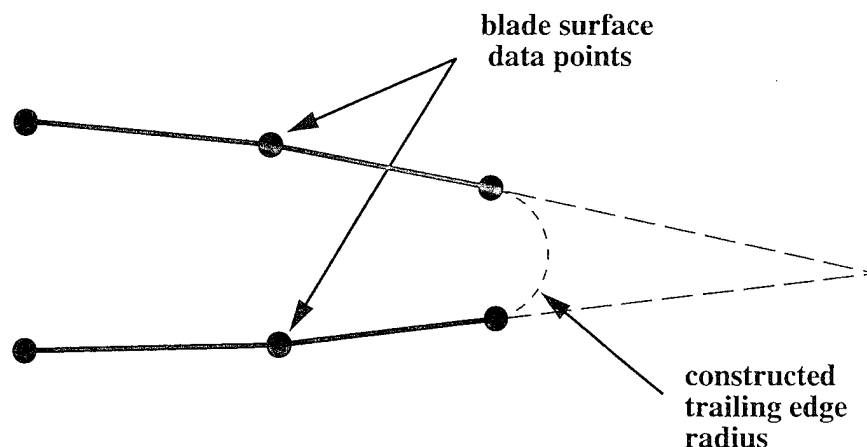
189

Figure C.2: Surface data used to construct blade trailing edge.

3-D surfaces which use a local coordinate system centered at the stacking axis of the blade. Typical UD0300 program output provides 80 points on each of the blade surfaces, with another 30 points used to describe the leading edge radius; the trailing edge is assumed to be blunt. The grid generator modifies the blunt trailing edge condition by fitting a circular arc within the wedge angle formed by the last surface segments on each of the two sides of the blade, as shown in Fig. C.2.

A second data file is used to position the blade shapes, each having their own local coordinate system, within the global coordinate system of a multi-blade row machine. This file is simply the meridional $(r-z)$ plane axisymmetric computational mesh which evolves in the streamline curvature algorithm of the UD0300 program; an example mesh for a transonic compressor is shown in Fig. C.3. This mesh describes a series of concentric, annular streamtubes through the machine and is arranged such that the mass flow is equal in each of the streamtubes and remains constant through the length of the machine. This results in a zero flow condition across the streamtube boundaries, which is one of the fundamental assumptions of the quasi-3D formulation. This global coordinate system file also allows multiple blade row

190
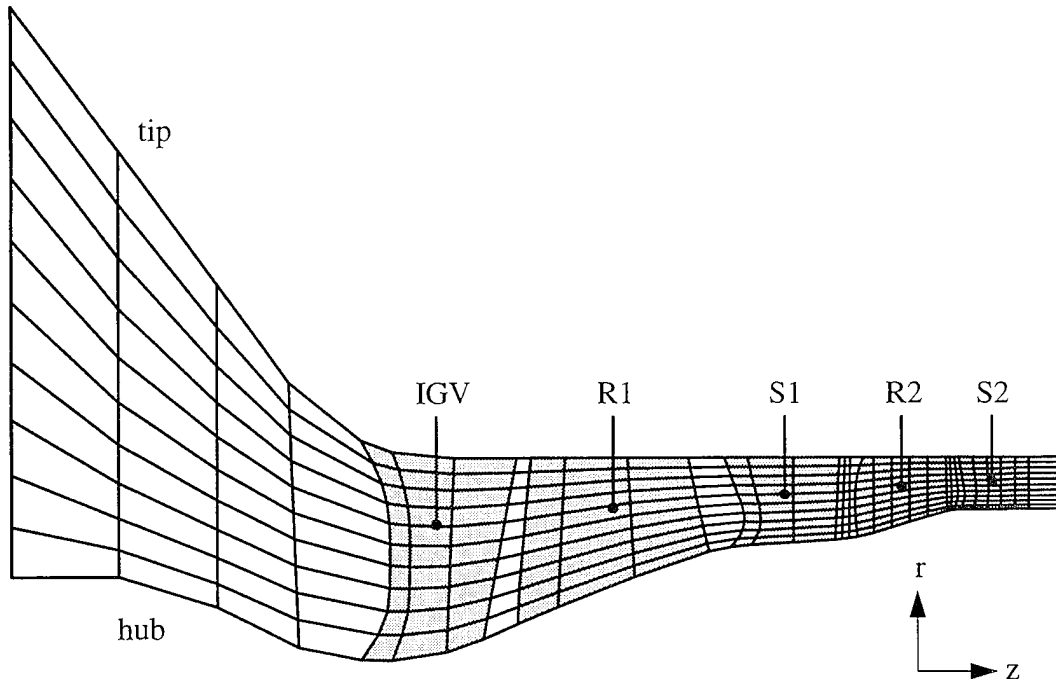
Figure C.3: Meridional mesh for a transonic compressor.

cases to be generated by providing the necessary positioning information for each of the separate blade surface files.

The streamtube thickness and radius variation information is generated by combining these two file types. For a grid developed along a particular streamsurface, the radius variation is taken from the radial values along the streamsurface; thickness values are determined by taking the difference in radius between the two adjacent streamsheets.

C.3 Grid Generation

The generation of the computational grid proceeds in two phases. The first is the definition of the computational boundaries and the various grid parameter, followed by a series of smoothing steps to achieve an acceptable final grid.

The computational domain is defined by the blade surfaces, the inlet and exit boundaries, and the periodic boundaries up- and downstream of the leading and

trailing edges respectively. The user has control of the length and angle of the up- and downstream sections, as well as the boundary between the H- and O-grid portions of the grid. Counter widgets on the graphical interface are used to set the number of grid lines in each direction of the various grid blocks. These counters are operated by repeated mouse clicks to increment or decrement to the desired value, or by typing in the number directly. The current value of the patch size is used when incrementing or decrementing counter values to insure that connectivity requirements are satisfied between the various grid blocks. Finally, slider-bar widgets are provided to set the coefficient of the exponential stretching functions applied along and normal to the blade surface, on the periodic boundaries, and across the passage between blades. The grid is updated with these new values in real-time, allowing the user to immediately see the impact of any changes in these variables.

Once the grid definition parameters are selected, an initial grid is constructed by simply connecting the appropriate points along the various boundaries. Because of the complexities of the geometry in a typical turbomachinery cascade, this grid is unacceptable for computation and must be smoothed. Two methods are used to smooth this initial grid, providing a grid with smoothly varying cell areas and minimal cell skewing. The first is the elliptic method of Sorenson [70], which may be applied throughout the entire H-grid portion of the grid or only in the center passage H-grid section. A simple Laplacian smoother is then available to uniformly smooth the entire grid. Each of these smoothing algorithms is available individually, or may be used in a combined mode with an iteration of each performed alternately. A typical grid smoothing session proceeds with a near convergence of the elliptic smoother followed by a number of Laplacian smoothing iterations to equalize the cell sizes along the H- and O-grid boundaries.

C.4   Graphical Interface

The gridding tool graphical interface was designed to allow the user to access virtually all of the parameters needed to generate the grid. Each of these quantities is available and may be changed at any point in the gridding procedure; any parameter change is immediately incorporated in the user's view of the grid. An example of the GUI is shown in Fig. C.4.

The GUI is divided functionally into five separate windows, or panels: the large grid window at the bottom left displays the current view of the grid during generation; the main control panel along the top is used to control the file I/O and contains the dials which are used to zoom and translate the grid in the grid window. This panel also contains buttons which determine the portions of the grid that are visible, and controls the application of the smoothing operators. The panel at the upper right is used to set the main geometrical parameters of the computational domain, such as the up- and downstream periodic boundary lengths and angles and the spacing between blade rows. Counters for the row number, number of passages in a particular blade row, and the patch size are also located here. The last two panels along the right side contain the counters and sliders for the grid dimensions and boundary stretching values for the H- and O-grid portions of the grid respectively. Since the edges of all blocks in the grid must contain an integer number of patches, the dimension counters are incremented or decremented by the current patch size.

The GUI has been implemented using the widget set from the "panel library" of the NASA-developed FAST post-processor [75]. Unfortunately, this library limits use of the grid generator to Silicon Graphics workstations; conversion of the interface to the Motif widget set is planned for the future.
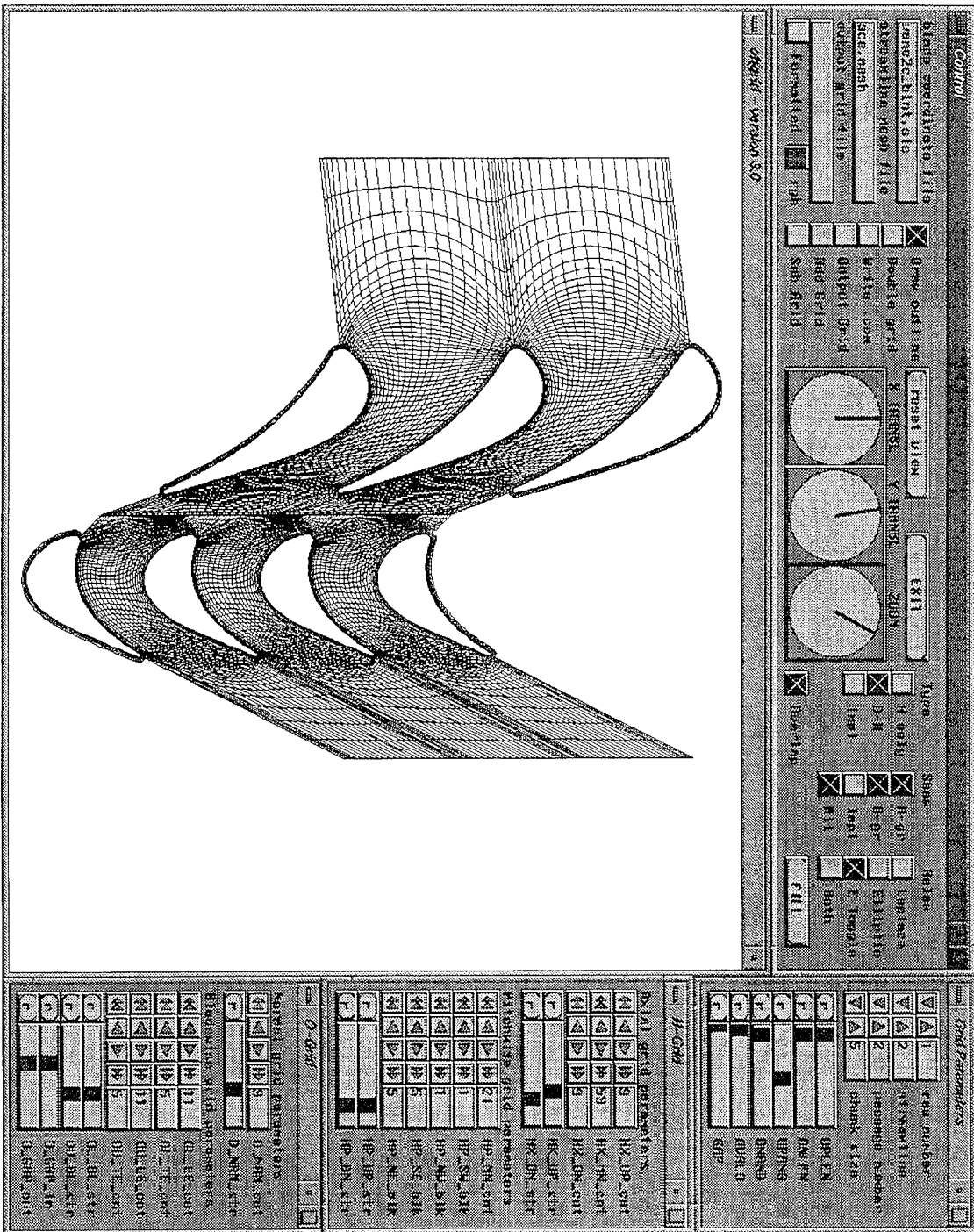
Figure C.4: Grid generator graphical user interface; 2:3 blade ratio turbine case shown.
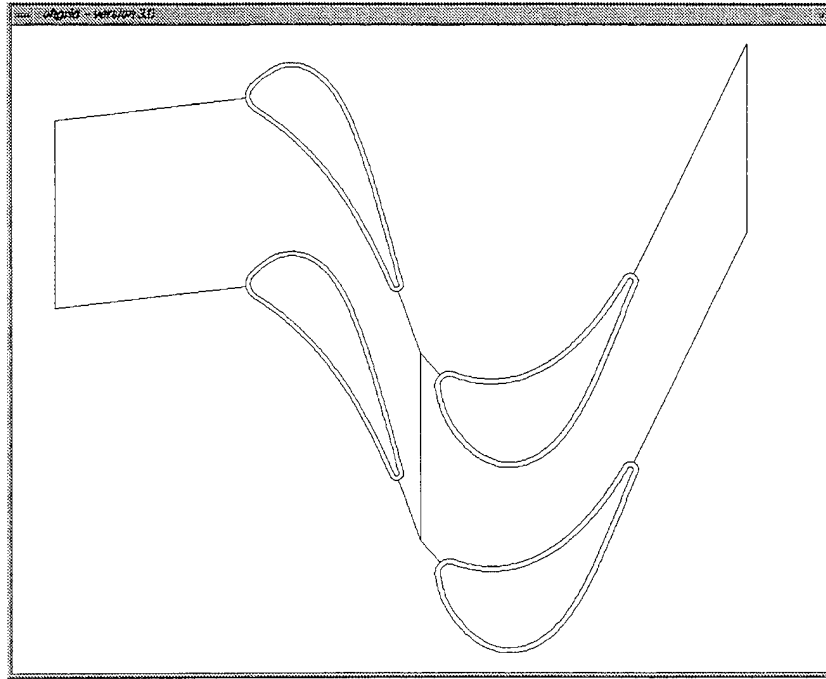
Figure C.5: Initial view of grid box.

C.5   Example Mesh Generation

In order to illustrate the use of the program, an example grid generation session is presented. This case is a two blade row transonic turbine stage with a 1:1 blade count ratio.

After entering the names of the meridional mesh and two blade streamline coordinate files, the user sees the "grid box," which consists of outlines of the blade surfaces and the edges of the computational domain. If the user is generating an O-H grid, the outer edge of the O-grid surrounding the blade surface is also displayed. As mentioned previously, sliders in the GUI allow the user to adjust the upstream and downstream grid boundary angles and lengths, the distance between blade rows, and the size of the O-grid oval around the blades. The grid window for this initial grid box view is shown in Fig. C.5.

The next step is to fill the grid boundaries with the unsmoothed initial grid. At this point, the user selects the number of grid points in each of the blocks of a multi-
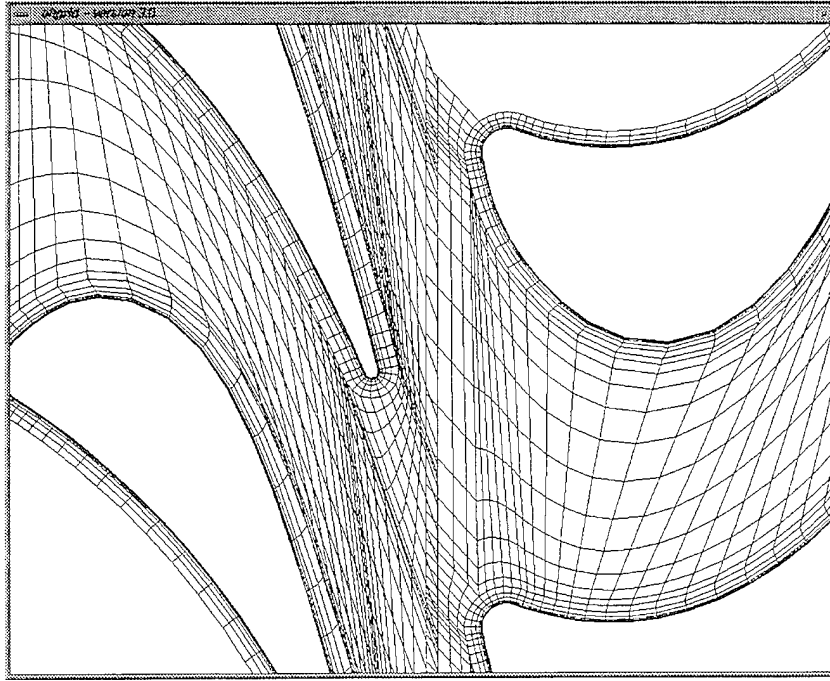
195

Figure C.6: Initial grid in "filled" state.

block grid, and the stretching functions applied along the various grid boundaries. Grid points are located along the boundaries in accordance with the selected count and stretching values and the connectivity requirements between the various grid blocks. This is termed the "filled state" and is shown in Fig. C.6.

This initial grid is now smoothed with the application of the elliptic and Laplacian smoothing functions. Typically, elliptic smoothing is first performed to achieve a nearly converged grid solution in the H-grid blocks. The Laplacian function is then applied to blend the interface between the O- and H-grid portions of the grid. The GUI contains buttons for each of these smoothers; each button-push causes one iteration of the selected smoothing algorithm to be performed. A close-up of the region between blade rows after the elliptic smoothing is shown in Fig. C.7; the final grid, after application of the Laplacian smoother, appears in Fig. C.8.

The final step in the process is the selection of an output filename and grid file type. The grid data is written in the format used by the Plot3D and FAST programs
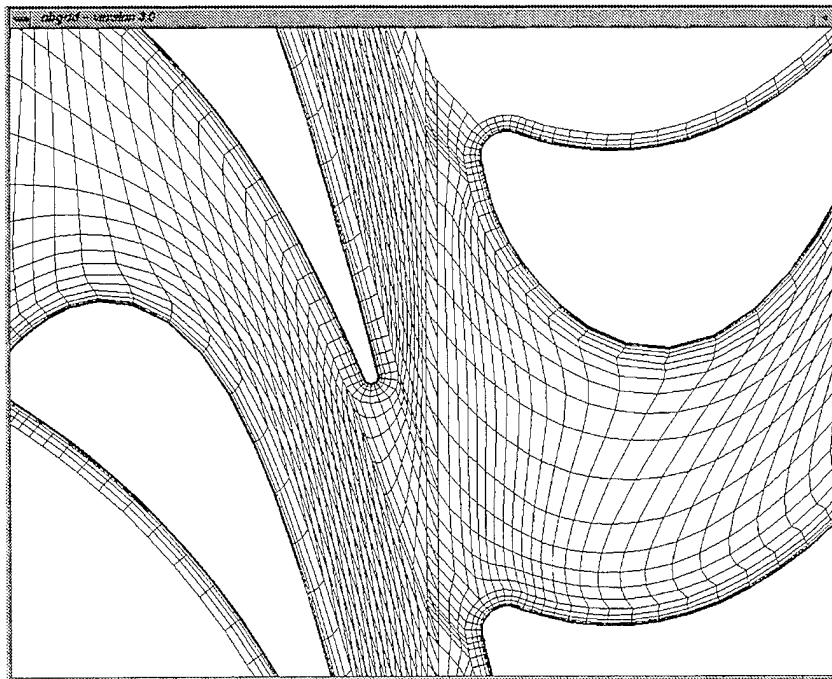
Figure C.7: Grid after elliptic smoothing.
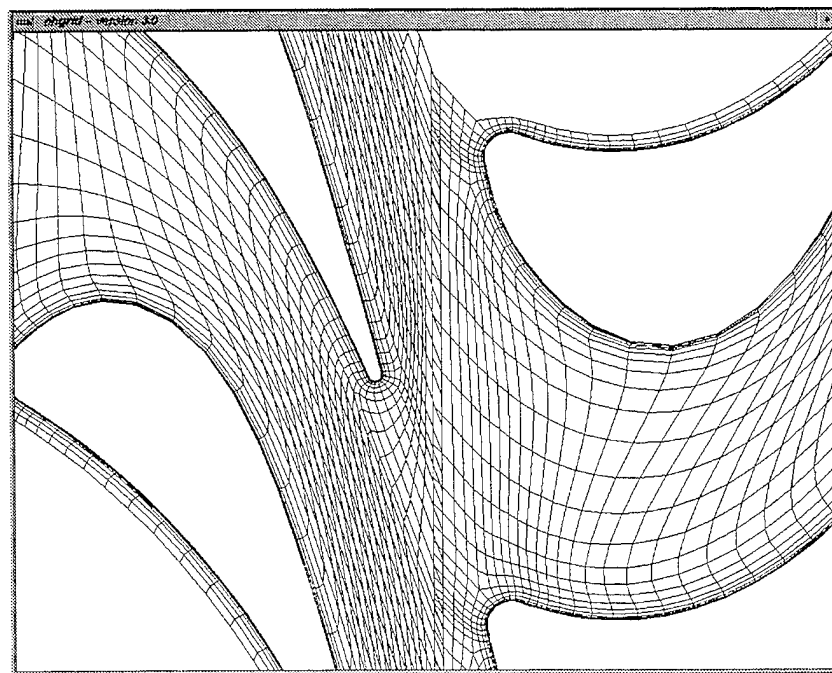
Figure C.8: Final grid after Laplacian smoothing.

[12, 74, 75] and may be written in either a formatted or unformatted form. The user is also given the option to write a small file containing all the parameters used to generate the grid in its pre-smoothed state. This file can be saved and read in later to restart the grid generation procedure. The grid output routine also writes the files which contain the radius and streamtube thickness variations, the blade surface spline data, and the multi-block connectivity information.

Bibliography

[1] R.S Abhari and A.H. Epstein. "An Experimental Study of Film Cooling in a Rotating Transonic Turbine". ASME Paper 92-GT-201, 1992.

[2] M.J Aftomis and N. Kroll. "A Quadrilateral Based Second-Order TVD Method for Unstructured Adaptive Meshes". AIAA Paper 91-0124, 1991.

[3] D.A. Anderson, J.C. Tannehill, and R.H. Pletcher. <u>Computational Fluid Mechanics and Heat Transfer</u>. Hemisphere Publishing, New York, 1984.

[4] A. Arnone, M.-S. Liou, and L.A. Povinelli. "Multigrid Time-Accurate Integration of Navier-Stokes Equations". AIAA Paper 93-3361-CP, 1993.

[5] A. Arnone and R. Pacciani. "Rotor-Stator Interaction Analysis Using the Navier-Stokes Equations and a Multigrid Method". ASME Paper 95-GT-177, 1995.

[6] D.A. Ashworth, J.E. LaGraff, D.L. Schultz, and K.J. Grindrod. "Unsteady Aerodynamic and Heat Transfer Processes in a Transonic Turbine Stage". *ASME Journal of Engineering for Gas Turbines and Power*, Vol. 107, pp. 1022-1030, 1985.

[7] B. Baldwin and H. Lomax. "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows". AIAA Paper 78-257, 1978.

[8] B.S. Baldwin and T.J. Barth. "A One-Equation Turbulence Transport Model for High Reynolds Number Wall-Bounded Flows". NASA TM-102847, 1990.

[9] T.J. Barth. "Aspects of Unstructured Grids and Finite Volume Solvers for the Euler and Navier-Stokes Equations". AGARD-R-787, 1992.

[10] P. Beaudan. "Numerical Experiments on the Flow Past a Circular Cylinder at Sub-Critical Reynolds Number". Ph.D. Thesis, Dept. of Mechanical Engineering, Stanford University, 1994.

[11] J.U. Brackbill and J.S. Saltzman. "Adaptive Zoning for Singular Problems in Two Dimensions". *Journal of Computational Physics*, Vol. 46, pp. 342-368, 1982.

[12] P. Buning and J.L. Steger. "Graphics and Flow Visualization in Computational Fluid Dynamics". AIAA Paper 85-1507, 1985.

[13] B. Cantwell and D. Coles. "An Experimental Study of Entrainment and Transport in the Turbulent Near Wake of a Circular Cylinder". *Journal of Fluid Mechanics*, Vol. 136, pp. 321-374, 1983.

[14] B. Carnahan, H.A. Luther, and J.O. Wilkes. Applied Numerical Methods. John Wiley & Sons, 1969.

[15] R.V. Chima. "Explicit Multigrid Algorithm for Quasi-Three-Dimensional Viscous Flows in Turbomachinery". *AIAA Journal of Propulsion and Power*, Vol. 3, pp. 397-405, 1987.

[16] T.J. Coakley. "Turbulence Modeling Methods for Compressible Navier-Stokes Equations". AIAA Paper 83-1693, 1983.

[17] R. Courant, K.O. Freidrichs, and H. Levy. "On the Partial Difference Equations of Mathematical Physics". *IBM Journal of Research and Development*, Vol. 11, pp. 215-234, 1967.

[18] N.A. Cumpsty. Compressor Aerodynamics. Longman Scientific and Technical, New York, 1989.

[19] J.F. Dannenhoffer III. "Grid Adaptation for Complex Two-Dimensional Transonic Flows". Sc.D. Thesis, Dept. of Aeronautics and Astronautics, MIT, 1987.

[20] C.B. Davies and E. Venkatapathy. "The Multidimensional Self-Adaptive Grid Code, SAGE". NASA TM-103905, 1992.

[21] R.L. Davis, D. Choi, and J.F. Dannenhoffer III. "Application of Two-Equation Turbulence Models with Adaptive Grid-Embedding Procedures". AIAA Paper 94-2716, 1994.

[22] R.L. Davis and J.F. Dannenhoffer III. "Three-Dimensional Adaptive Grid Embedding Euler Technique". AIAA Paper 93-0330, 1993.

[23] W.N. Dawes. "A Numerical Study of the Interaction of a Transonic Compressor Rotor Overtip Leakage Vortex with the Following Stator Blade Row". ASME Paper 94-GT-156, 1994.

[24] J.D. Denton. "An Improved Time Marching Method for Turbomachinery Flow Calculation". ASME Paper 82-GT-239, 1982.

[25] S. Fleeter, R.L. Holtman, R.B. McClure, and G.T. Sinnet. "Experimental Investigation of a Supersonic Compressor Cascade". ARL TR 75-0208, Aerospace Research Laboratories, Wright-Patterson AFB, OH, 1975.

[26] M.B. Giles. "Calculation of Unsteady Wake/Rotor Interaction". AIAA Paper 87-0006, 1987.

[27] M.B. Giles. "Non-Reflecting Boundary Conditions for the Euler Equations". Technical Report TR-88-1, MIT Computational Fluid Dynamics Laboratory, 1988.

[28] M.B. Giles. "UNSFLO: A Numerical Method For Unsteady Inviscid Flow In Turbomachinery". GTL Technical Report 195, MIT Gas Turbine Laboratory, 1988.

[29] M.B. Giles and R. Haimes. "Validation of a Numerical Method for Unsteady Flow Calculations". *ASME Journal of Turbomachinery*, Vol. 115, pp. 110-117, 1993.

[30] K.L. Gundy-Burlet, M.M. Rai, and R.C. Stauter. "Temporally and Spatially Resolved Flow in a Two-Stage Axial Compressor: Part 2 - Computational Assessment". *ASME Journal of Turbomachinery*, Vol. 113, No. 2, 1991.

[31] H.P. Hodson. "An Inviscid Blade-to-Blade Prediction of a Wake-Generated Unsteady Flow". ASME Paper 84-GT-43, 1984.

[32] A. Jameson. "Time Dependent Calculations Using Multigrid, with Application to Unsteady Flows Past Airfoils and Wings". AIAA Paper 91-1596, 1991.

[33] C.E. Jobe and W.L. Hankey. "Turbulent Boundary-Layer Calculations in Adverse Pressure Gradient Flows". *AIAA Journal*, Vol. 18, No. 11, 1980.

[34] D.A. Johnson and L.S. King. "A Mathematically Simple Turbulence Closure Model for Attached and Separated Turbulent Boundary Layers". *AIAA Journal*, Vol. 23, pp. 1684-1692, 1985.

[35] W.P. Jones and B.E. Launder. "The Prediction of Laminarization with a Two-Equation Model of Turbulence". *International Journal of Heat and Mass Transfer*, Vol. 15, pp. 301-314, 1972.

[36] P.C.E. Jorgenson and R.V. Chima. "An Explicit Runge-Kutta Method for Unsteady Rotor/Stator Interaction". AIAA Paper 88-0049, 1988.

[37] I. Kallinderis. "Adaptation Methods for Viscous Flows". Ph.D. Thesis, Dept. of Aeronautics and Astronautics, MIT, 1989.

[38] S. Ko and W.J. McCroskey. "Computations of Unsteady Separating Flows Over an Oscillating Airfoil". AIAA Paper 95-0312, 1995.

[39] N. Kroll and C. Rossow. "Foundations of Numerical Methods for the Solution of Euler Equations". DLR Lecture Series F6.03, Braunschweig, Germany, 1989.

[40] C.H. Law and A.R. Wadia. "Low Aspect Ratio Transonic Rotor: Part 1 - Baseline Design and Performance". *ASME Journal of Turbomachinery*, Vol. 115, pp. 218-225, 1993.

[41] J.P. Lewis, Hall. E.J., and R.A. Delaney. "Numerical Prediction of Turbine Vane-Blade Aerodynamic Interaction". *ASME Journal of Turbomachinery*, Vol. 111, pp. 387-393, 1989.

[42] S.J. Lin and Yang. R.J. "Viscous Unsteady Computations of Rotor/Stator Flows in the Multi-stage Turbines". AIAA Paper 91-2465, 1991.

[43] R.W. MacCormack and B.S. Baldwin. "A Numerical Method for Solving the Navier-Stokes Equations with Application to Shock-Boundary Layer Interactions". AIAA Paper 75-1, 1975.

[44] S.R. Mathur, N.K. Madavan, and R.G. Rajagopalan. "A Hybrid Structured-Unstructured Grid Method for the Unsteady Turbomachinery Flow Computations". AIAA Paper 93-0387, 1993.

[45] S.R. Mathur, N.K. Madavan, and R.G. Rajagopalan. "A Solution-Adaptive Hybrid Grid Method for the Unsteady Analysis of Turbomachinery". AIAA Paper 93-3015, 1993.

[46] D.J. Mavriplis. "Algebraic Turbulence Modeling for Unstructured and Adaptive Meshes". AIAA Paper 90-1653, 1990.

[47] D.J. Mavriplis and A. Jameson. "Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes". *AIAA Journal,* Vol. 28, pp. 1415-1425, 1990.

[48] F.R. Menter. "Improved Two-Equation $k - \omega$ Turbulence Models for Aerodynamic Flows". NASA TM 103975, 1992.

[49] N.A. Mitchell. "A Time-Marching Method for Steady Two-Dimensional Flow in a Blade Passage". *Int. J. Heat and Fluid Flow,* Vol.2, No. 4., 1980.

[50] J. Peraire, M. Vahdati, K. Morgan, and O.C. Zienkiewicz. "Adaptive Remeshing for Compressible Flow Computation". *Journal of Computational Physics,* Vol. 72, pp. 449-466, 1987.

[51] R. Radespiel and R.C. Swanson. "An Investigation of Cell Centered and Cell Vertex Multigrid Schemes for Navier-Stokes Equations". AIAA Paper 89-0543, 1989.

[52] M.M. Rai. "Navier-Stokes Simulations of Rotor-Stator Interaction Using Patched and Overlaid Grids". *AIAA Journal of Propulsion and Power,* Vol. 3, pp. 387-396, 1987.

[53] M.M. Rai. "Three-Dimensional Navier-Stokes Simulations of Turbine Rotor-Stator Interaction; Part I - Methodology". *AIAA Journal of Propulsion and Power,* Vol. 5, pp. 305-311, 1989.

[54] M.M. Rai. "Three-Dimensional Navier-Stokes Simulations of Turbine Rotor-Stator Interaction; Part II - Results". *AIAA Journal of Propulsion and Power,* Vol. 5, pp. 312-319, 1989.

[55] M.M. Rai and R.P. Dring. "Navier-Stokes Analyses of Redistribution of Inlet Temperature Distortions in a Turbine". *AIAA Journal of Propulsion and Power*, Vol. 6, pp. 276-282, 1990.

[56] K.V. Rao and R.A. Delaney. "3-D Simulation of Vane-Blade Interaction in a Transonic Turbine". AIAA Paper 93-2256, 1993.

[57] K.V. Rao, R.A. Delaney, and M.G. Dunn. "Vane-Blade Interaction in a Transonic Turbine, Part I - Aerodynamics". *AIAA Journal of Propulsion and Power*, Vol. 10, pp. 305-311, 1994.

[58] K.V. Rao, R.A. Delaney, and M.G. Dunn. "Vane-Blade Interaction in a Transonic Turbine, Part II - Heat Transfer". *AIAA Journal of Propulsion and Power*, Vol. 10, pp. 312-317, 1994.

[59] S.M. Richardson. "A Numerical Study of Unsteady Flow Effects in a Supersonic Compressor Cascade". AGARD CP-401, 1986.

[60] S.M. Richardson. "Analysis of Unsteady Rotor-Stator Interactions Using a Viscous Explicit Method". AIAA Paper 90-0342, 1990.

[61] S.M. Richardson. "An Unstructured Adaptive Quadrilateral Mesh-Based Scheme for Viscous Turbomachinery Flow Calculations". AIAA Paper 93-1975, 1993.

[62] S.M. Richardson. "Transonic Turbomachinery Calculations Using a Hybrid Structured-Unstructured Grid Method". ASME Paper 94-GT-62, 1994.

[63] A. Roshko. "On the Development of Turbulent Wakes from Vortex Streets". NACA TN 2913, 1953.

[64] A. Roshko. "On the Drag and Shedding Frequency of Two-Dimensional Bluff Bodies". NACA TN 3169, 1954.

[65] H. Schlichting. <u>Boundary Layer Theory</u>. McGraw-Hill Book Company, New York, 1979.

[66] J.N. Scott and W.L. Hankey. "Navier-Stokes Solutions of Unsteady Flow in a Compressor Rotor". ASME Paper 86-GT-226, 1986.

[67] J.S Shang, P.G. Buning, W.L. Hankey, and M.C. Wirth. "Performance of a Vectorized Three-Dimensional Navier-Stokes Code on the CRAY-1 Computer". *AIAA Journal*, Vol. 18, pp. 1073-1079, 1980.

[68] J.S Shang, W.L. Hankey, and J.S. Petty. "Three-Dimensional Supersonic Interacting Turbulent Flow along a Corner". *AIAA Journal*, Vol. 17, pp. 706-713, 1978.

[69] G.L. Siden, W.N. Dawes, and P.J. Albraten. "Numerical Simulation of the Two-Dimensional Viscous Compressible Flow in Blade Cascades Using a Solution-Adaptive Unstructured Mesh". *ASME Journal of Turbomachinery*, Vol. 112, No. 3, 1990.

[70] R.L. Sorenson. "A Computer Program to Generate Two-Dimensional Grids About Airfoils and Other Shapes by Use of Poisson's Equation". NASA TM-81198, 1980.

[71] P.R Spalart and S.R. Allmaras. "A One-Equation Turbulence Model for Aerodynamic Flows". AIAA Paper 92-0439, 1992.

[72] R.C. Swanson, E. Turkel, and J.A. White. "An Effective Multigrid Method for High-Speed Flows". ICASE Report 91-56, 1991.

[73] E.R. van Driest. "Turbulent Boundary Layers in Compressible Fluids". *Journal of the Aeronautical Sciences*, Vol. 18, pp. 145-160, 1951.

[74] P.P Walatka, P.G. Buning, L. Pierce, and P.A. Elson. "Plot3D User's Guide". NASA TM 10167, 1992.

[75] P.P Walatka, J. Clucas, R.K. McCabe, T. Plessel, and R. Potter. "FAST User Guide". NASA RND-93-010, 1993.

[76] J.M. Weiss and W.A. Smith. "Preconditioning Applied to Variable and Constant Density Time-Accurate Flows on Unstructured Meshes". AIAA Paper 94-2209, 1994.

[77] A.J. Wennerstrom and S.L. Puterbaugh. "A Three-Dimensional Model for the Prediction of Shock Losses in Compressor Rotor Blade Rows". *ASME Journal of Engineering for Gas Turbines and Power*, Vol. 106, pp. 295-299, 1984.

[78] F.M. White. Viscous Fluid Flow. McGraw-Hill Book Company, New York, 1974.

[79] D.C Wilcox. "Reassessment of the Scale-Determining Equation for Advanced Turbulence Models". *AIAA Journal*, Vol. 18, pp. 1299-1310, 1988.

[80] C.H. Wu. "A General Through-Flow Theory of Fluid Flow with Subsonic or Supersonic Velocity in Turbomachines of Arbitrary Hub and Casing Shapes". NACA TN 2302, 1951.

[81] C.H. Wu. "A General Theory of Three-Dimensional Flow in Subsonic and Supersonic Turbomachines of Axial, Radial and Mixed Flow Types". NACA TN 2604, 1952.